

## RAPPORT DE PROJET

### Master Professionnel INFORMATIQUE et ROBOTIQUE Parcours *"Expert du Numérisation et Innovation" (ENI)*

*Web ARSn- Applicatifs de supervision et contrôle*

The screenshot displays a web-based monitoring and control system for industrial robots. On the left, a sidebar lists navigation links: Page Voiture, Page MEF, Page PAV, Page PAR, and Page AGV. The main content area is divided into several sections:

- Alarme 1:** A configuration panel for alarm 1, showing triggers for "Déclenchement by day of the week" (Monday to Sunday) and "Déclenchement by month of the year" (January to December). It includes tabs for "Graphique", "Valors", and "RETOUR".
- Historisation:** A table showing historical data for alarms, with columns for alarm\_id, id\_code, nom, and déclenchement. The data includes entries for various robot faults and communication errors.
- Filtres/ Journalisation:** A section for filtering and journalizing data.
- Footer:** Includes copyright information (2024 Stellantis-ARSn. Tous droits réservés.) and a note about cookie usage.

*LOTODE Morgan*

Tuteur Entreprise :  
*FORNEROD Antoine*  
Tuteur Pédagogique  
*LEMANE Éric*



Entreprise d'Accueil :  
*Société A.R.S.N*  
*11 rue des Orchidées*  
*35650 LE RHEU*  
*Tel : 02-995249-00*

Année 2023-2024



# Rapport de Projet

## Applicatifs de supervision & contrôle ARSn

-  
Maquettage & Prototype destiné à l'industrie Automobile

*LOTODE Morgan*

Révision du 04/04/2024

## GESTION DES RÉVISIONS

Révision	Objet de la révision	Rédaction	Date
1	Création du document	M.LOTODE	05/03/2024
2	Révision 1	M.LOTODE	28/03/2024
3	Révision 2	M.LOTODE	02/04/2024
4	Révision Finale	M.LOTODE	04/04/2024
5			

## 1 REMERCIEMENTS

Je tiens tout d'abord à exprimer ma profonde gratitude envers M. BOUSSADE, Président et actionnaire chez ARSn.

Leur engagement envers mon développement professionnel a été essentiel tout au long de mon année d'alternance.

Je souhaite également adresser mes remerciements à mes nouveaux collègues chez ARSn pour leur accueil chaleureux, leur esprit d'équipe et les connaissances précieuses qu'ils m'ont transmis. En particulier, je suis reconnaissant envers mon tuteur d'entreprise, M. FORNEROD, pour sa guidance et son partage de savoir, qui ont grandement enrichi mon expérience professionnelle dans le domaine de la robotique automobile.

Je n'oublie pas l'apport précieux de mes enseignants, M. LEMANE et M. MEDJEROUB, ainsi que le centre de formation de l'UIMM dans son ensemble, pour leur engagement et leur professionnalisme qui ont contribué à mon apprentissage et à ma progression dans le cadre de mon diplôme CDA.

Malgré mon changement d'entreprise au cours de mon cursus, je remercie Robot-System. Je tiens à exprimer ma reconnaissance envers Mme. BOURDAIS-GALMARD, M. PERVIS, ainsi que toute l'équipe pour m'avoir accueilli et accompagné lors de mon expérience précédente. Leurs conseils, leur soutien et les opportunités offertes m'ont permis de développer des compétences précieuses dans le domaine de la robotique.

Enfin, un immense merci à ma famille, mes ami·es et toutes les personnes qui ont contribué, de près ou de loin, au succès de mon année d'alternance chez ARSn. Leurs encouragements et leurs conseils ont été inestimable dans mon parcours.

Merci à toutes les personnes qui ont contribué au succès de cette année d'alternance

## SOMMAIRE

### Table des matières

1 REMERCIEMENTS.....	4
2 ABSTRACT .....	9
3 INTRODUCTION .....	10
4 PRÉSENTATION DE L'ENTREPRISE & DU SERVICE.....	11
4.1 ARSN c'est Qui ? .....	11
4.2 Secteurs de l'entreprise.....	12
4.3 Activités de l'entreprise.....	12
4.4 Des clients satisfaits .....	13
4.5 Service .....	14
4.6 Organigramme.....	14
5 LISTE DU PROJET.....	15
6 MOTS-CLÉS REPRÉSENTATIFS DU PROJET.....	15
7 PRÉSENTATION DU SYSTÈME .....	16
7.1 Raisons du projet.....	16
7.2 Présentation client .....	16
7.3 Détail des fonctionnalités du système actuel .....	18
8 PRÉSENTATION DU PROJET .....	19
8.1 Fonctionnalité Générale .....	19
9 FLANIFICATION & GESTIONS DES TACHES .....	21
9.1 GANTT Simplifié.....	21
9.2 Méthode Agile .....	21
9.3 Méthode versionnage projet.....	22
10 FONCTIONS DÉTAILLÉES .....	23
1. AGV .....	23
2. Dévracage Vis.....	23
3. Robot Vissage.....	23
4. Poste re taquage.....	23
5. Poste tampon.....	23

6.	Robot Vision.....	23
7.	Claye d'approvisionnement Porte .....	23
11	CYCLE SIMPLIFIÉ .....	24
12	LIGNE ILLUSTRÉE.....	24
13	PRESENTATION DES PRODUITS .....	25
13.1	Pièces Primaires .....	25
13.2	Pièce Finie .....	25
14	DESCRIPTIF DE LA LIGNE.....	26
14.1	Fonctionnalités de l'installation actuelle (Robotisation simple).....	26
14.2	Fonctionnalités de l'installation proposé (Persistance de données) .....	26
15	ÉCHANGES & DONNÉES.....	27
15.1	Échanges entre les différents clients .....	27
15.2	Diagramme de pieuvre .....	28
15.3	Fonctions Principales .....	29
15.4	Allocation .....	30
16	RÔLES & FONCTIONNALITÉES.....	30
16.1	Operateur.....	30
16.2	Conducteur de ligne.....	30
16.3	Invité / Commercial.....	30
16.4	Administrateur.....	30
17	CAS D'UTILISATION DES APPLICATIFS.....	31
18	DICTIONNAIRE DE CLASSES / DONNÉES .....	32
19	MODÉLISATION DE LA BDD .....	33
19.1	Méthode MCD.....	33
19.1.1	Processus d'attaque.....	33
19.1.2	Processus d'affichage.....	33
19.2	Modélisation de la BDD méthode MLD .....	34
20	MAQUETTAGE WEB.....	35
21	ARCHITECTURE MATERIEL.....	36
21.1	Architecture réseau du système .....	36
21.2	Architecture réseau du système .....	37
21.3	Choix du matériel.....	38

21.4	Choix des logiciels .....	38
21.5	Principaux langages .....	38
22	DÉVELOPPEMENT DU PROJET WEB.....	39
22.1	Application Web (Flask) .....	39
	Qu'est-ce que c'est ? : .....	39
22.2	Programmation Application Web .....	40
22.2.1	Structure Applicatif Leger .....	40
22.2.2	Environnements virtuels.....	40
22.2.3	Variables d'environnement .....	40
22.2.4	Requêtes préparées ORM.....	40
22.3	Recommandation sécurité OWASP – ANSSI .....	41
23	DÉVELOPPEMENT DU PROJET LOURD.....	42
23.1	Application Lourde (Pyqt5) .....	42
23.2	Programmation Application Lourde .....	43
23.2.1	Structure Applicatif Lourd.....	43
23.2.2	Fonction de mouvement Robot.....	43
23.2.3	Fonction Widget / Interface Utilisateur.....	44
23.3	Simulation et implantation mécanique .....	45
23.4	Table d'échanges .....	46
23.5	Communication entre Applicatif Lourd - Robot .....	46
23.6	Programmation du Robot conformisation /vissage .....	47
23.6.1	Programme principal d'aiguillage .....	47
23.6.2	Programme de mouvement.....	47
24	PLAN DE TESTS DES APPLICATIFS .....	48
24.1	Plan de tests application lourde .....	48
24.2	Plan de tests application légère.....	48
25	TESTS DE L'APPLICATIF WEB DE PRE-DEPLOIEMENT .....	49
24.3	Tests Unitaires .....	49
24.4	Test d'Intégration.....	50
24.5	Test Fonctionnel.....	51
26	DEPLOYEMENT DES APPLICATIFS .....	52
26.1	Plan de déploiement Applicatif Web .....	52

26.1.1	Préparation .....	52
26.1.2	Déploiement .....	52
26.1.2.1	Dockerisation applicatif web .....	52
26.1.3	Post-Déploiement .....	55
27	TESTS DE L'APPLICATIF WEB POST-DEPLOIEMENT .....	56
26.1	Test de charge.....	56
26.2	Tests de performance .....	57
26.3	Tests de sécurité & vulnérabilités.....	58
28	FINALITÉ DU PROJET WEB APP ARSn.....	59
29	COMPÉTENCES ACQUISES .....	60
30	CONCLUSION .....	61
31	LISTE DES FIGURES.....	62
32	ANNEXES.....	64

## 2 ABSTRACT

At ARSn, we specialize in handling machines from conception to final product, offering comprehensive services such as pre-project advice, robotics simulation, feasibility tests, and machine diagnostics.

Our expertise extends to retrofitting and preventive maintenance, ensuring optimal performance and longevity of our systems. With a focus on robotics, we cover a wide range of configurations and brands, including Scara and Delta robots, and incorporate features like tracking, recognition, and quality checks into our machines.

While our primary focus at ARSn currently lies in automotive robotics, we remain committed to delivering advanced automation solutions across various industries. The automotive sector demands precision, efficiency, and reliability, and our projects are tailored to meet these exacting standards. From assembly line automation to quality control systems, we provide bespoke solutions to address the specific needs of each client.

As part of my role at ARSn, I am frequently required to travel both domestically and internationally to oversee project implementation and provide technical support.

This exposure to diverse environments and cultures not only enhances my professional experience but also underscores the global impact of our robotics solutions. It is a testament to our dedication to excellence and innovation in the field, driving us to continually push the boundaries of what is achievable in automotive automation and beyond.

---

Chez ARSn, nous nous spécialisons dans la manipulation des machines de la conception au produit final, offrant des services complets tels que des conseils avant-projet, la simulation robotique, les tests de faisabilité et le diagnostic des machines.

Notre expertise s'étend à la modernisation et à la maintenance préventive, assurant une performance et une longévité optimales de nos systèmes. En mettant l'accent sur la robotique, nous couvrons un large éventail de configurations et de marques, y compris les robots Scara et Delta, et intégrons des fonctionnalités telles que le suivi, la reconnaissance et les contrôles de qualité dans nos machines.

Bien que notre objectif principal chez ARSn réside actuellement dans la robotique automobile, nous restons engagés à fournir des solutions d'automatisation avancées dans divers secteurs. Le secteur automobile exige précision, efficacité et fiabilité, et nos projets sont conçus pour répondre à ces normes rigoureuses. De l'automatisation des chaînes de montage aux systèmes de contrôle qualité, nous fournissons des solutions sur mesure pour répondre aux besoins spécifiques de chaque client.

Dans le cadre de mes fonctions chez ARSn, je suis fréquemment appelé à voyager au pays et à l'étranger pour superviser la mise en œuvre du projet et fournir un soutien technique.

Cette exposition à divers environnements et cultures améliore non seulement mon expérience professionnelle, mais souligne également l'impact mondial de nos solutions robotiques. Cela témoigne de notre engagement envers l'excellence et l'innovation dans le domaine, ce qui nous pousse à repousser continuellement les limites de ce qui est réalisable dans l'automatisation automobile et au-delà.

### 3 INTRODUCTION

Pendant ma première année de Licence ISAR (Ingénierie des Systèmes Automatiques et Robotiques) à l'IUT de Rennes 1, j'ai pu acquérir des compétences dans différents domaines tels que la mécatronique et la robotique. Ce dernier m'a particulièrement passionné. J'ai eu l'opportunité de participer aux Olympiades FANUC de 2021 en tant que participant, ainsi qu'à celles de 2023 et 2024 en tant que membre de l'équipe projet.

J'ai donc décidé de poursuivre mes études en réalisant le master professionnel en Robotique à l'IUMM de Bruz, dans le but d'acquérir les compétences nécessaires pour devenir informaticien/ roboticien.

Cette formation se déroulant en alternance et dans le but de concrétiser mon projet professionnel, j'ai choisi de rejoindre ARSn, une entreprise riche de 20 années d'expérience dans les domaines de la robotique, de l'automatisme du secteur de l'automobile.

Le fait d'effectuer cette formation en alternance permet également de s'adapter au fonctionnement d'une entreprise et d'appliquer les connaissances développées durant les périodes de formation à l'IUMM.

ARSn s'est forgée une solide réputation grâce à la qualité de ses machines intégrant automatiquement des robots. Elle conçoit des machines spéciales et intervient dans plusieurs domaines d'activité tels que le bois, la métallurgie, l'agroalimentaire, la pharmaceutique et la plasturgie. ARSn dispose également d'un pôle spécialisé dans la prestation automobile pour de grands clients, où les techniciens sont amenés à se déplacer en France, en Europe et dans le monde entier. Ce pôle consiste à intégrer des lignes robotisées, souvent composées de plusieurs robots.

L'objectif de l'entreprise est de développer des lignes flexibles capables de traiter des produits dans diverses conditions de processus.

Tout au long de l'année, j'ai travaillé sur ces systèmes, que ce soit pour des applications métier de soudure, de découpe laser, de manutention, de palettisation, de ferrage, de simulation de processus, de mise en service et d'installation sur site.

Mon projet principal a été d'étudier et de développer une partie d'une cellule composée de 4 robots poly articulés : la "ligne de montage MEF Porte avant". Cette cellule vise à réaliser une fonction spécifique au sein du système d'assemblage et de vissage de portes sur une carrosserie de voiture. J'ai également conceptualisé et réalisé une application web de supervision et de contrôle pour détecter les défauts et prendre des décisions, ainsi que pour envoyer des ordres aux robots.

La cellule intégrée doit assembler les portes sur la carrosserie pour former la voiture. Deux robots déposent le produit dans des caisses selon des formats spécifiques avec l'aide de la vision. Deux autres robots visent la porte à la carrosserie avec quatre vis de sécurité. Les caisses sont apportées par des AGV dans l'îlot, tandis que les produits (portes) sont amenés par des caristes par lot de 20, et les visseries sont déposées et acheminées par un bol vibrant.

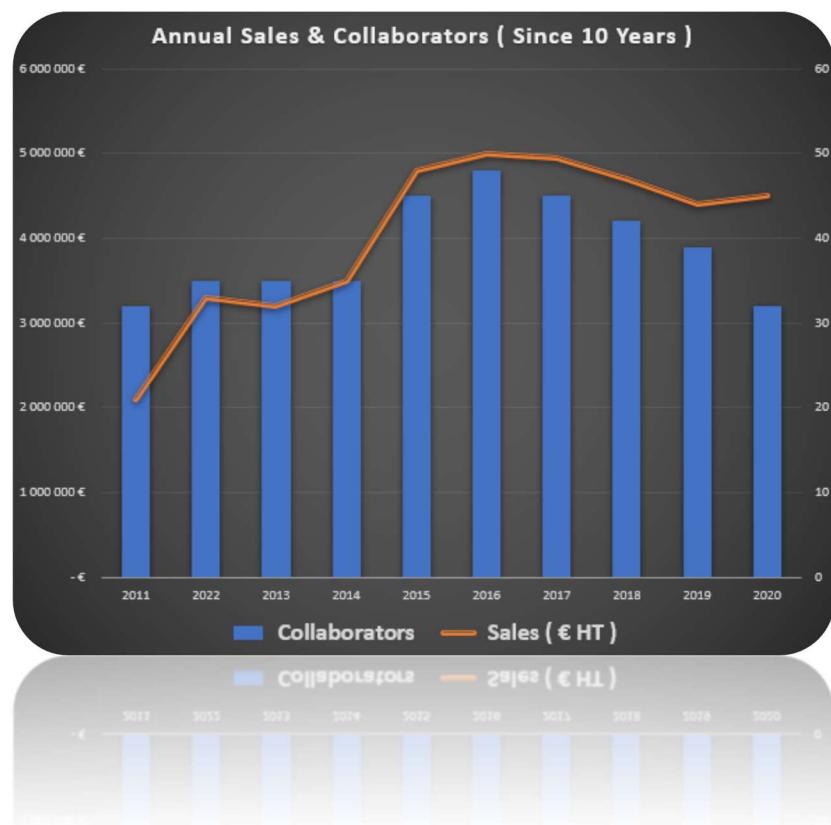
Dans ce rapport, nous définirons les raisons du projet et expliquerons le fonctionnement de la cellule du système, en mettant particulièrement l'accent sur la cellule de vissage. Nous détaillerons ensuite, les missions ainsi que les choix qui ont été faits pour mener à bien ce projet. Dans un premier temps, nous présenterons, forte de son expérience, l'entreprise ARSn dans son ensemble, et plus spécifiquement le pôle de prestation automobile.

## 4 PRÉSENTATION DE L'ENTREPRISE & DU SERVICE

### 4.1 ARSN c'est Qui ?

L'entreprise ARSn a été créée en 2000 par HAMRAOUI Boussade. ARSn est une société d'ingénierie et de robotique basée en Bretagne, en France. L'entreprise possède plus de 20 ans d'expérience dans ce domaine et propose un large éventail de services liés aux robots industriels.

Depuis sa création, ARSn a connu une croissance constante, comme le montre le graphique ci-dessous (image fournie). En 2023, ARSn a réalisé un chiffre d'affaires de 4 millions d'euros et employait 30 collaborateurs.



*Figure 1 : La frise sales/collaborators depuis 10 ans*

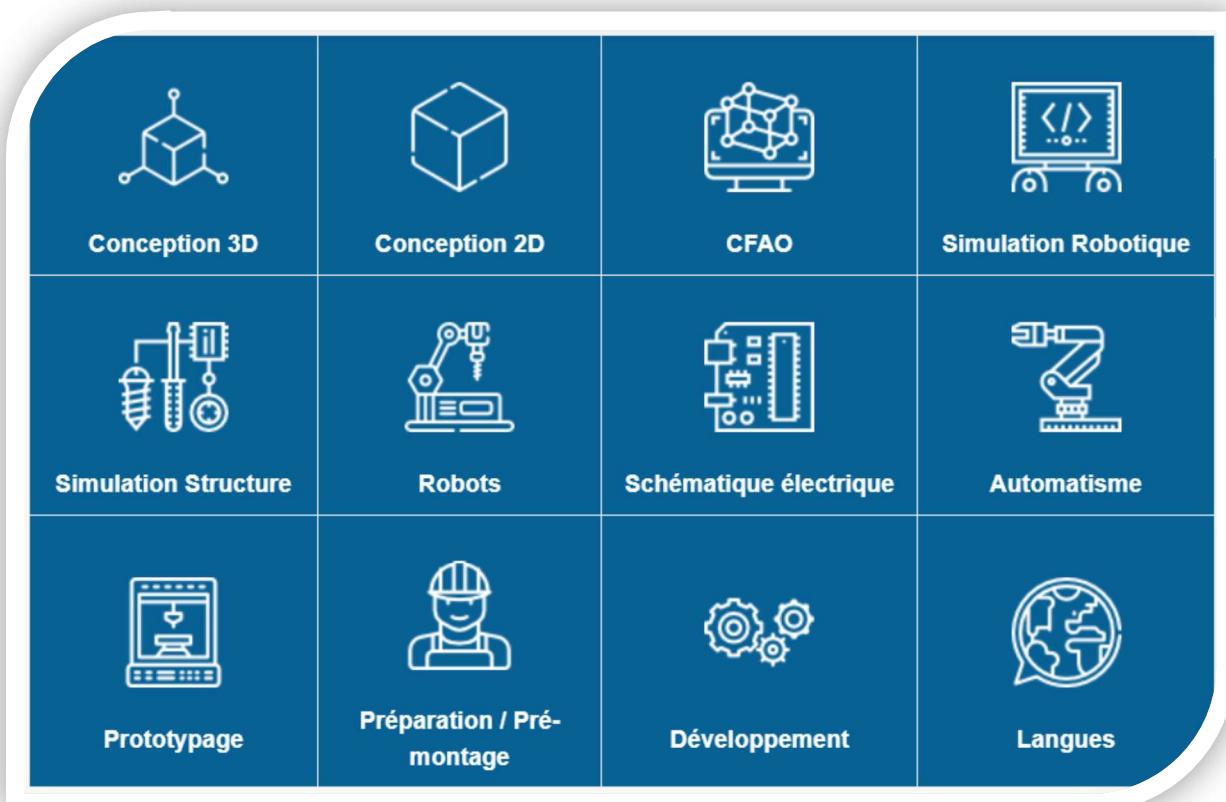
Le graphique ci-joint permet de visualiser la croissance de l'entreprise ARSn au cours des 10 dernières années. On y voit que le nombre de collaborateurs a triplé et que le chiffre d'affaires a été multiplié par 4.

#### Chiffres clés :

- 20 ans d'expérience dans le domaine de la robotique
- 4 millions d'euros de chiffre d'affaires en 2023
- 30 collaborateurs
- Plus de 100 clients en France et à l'international

## 4.2 Secteurs de l'entreprise

L'activité de l'entreprise couvre plusieurs secteurs comme la Métallurgie, la Plasturgie, le Bois & Papier, ou encore l'Agro-Alimentaire. Elle est spécialisée dans la conception de systèmes automatisés dotés d'au moins un robot et d'une caméra vison. Chaque machine est différente et répond aux attentes du cahier des charges client. En effet, l'objectif de Robot-System est de concevoir des machines spéciales « très flexibles » et de « forte cadence » capables de s'adapter à un grand nombre de références produits. Ceci rend l'entreprise très malléable et fiable, car elle est capable de s'adapter aux demandes de chaque client.



*Figure 2 : Les secteurs & domaines d'activités*

## 4.3 Activités de l'entreprise

ARSn propose une gamme complète de services liés à la robotique, du conseil initial à la maintenance en passant par la conception, la réalisation et l'intégration. L'entreprise peut ainsi prendre en charge l'intégralité d'un projet robotique ou fournir des services spécifiques en fonction des besoins du client.

ARSn s'engage à écouter attentivement les besoins de ses clients afin de proposer des solutions innovantes et parfaitement adaptées à leurs exigences. L'entreprise s'appuie sur son expertise en ingénierie et en robotique pour développer des machines uniques et performantes.

Lorsque cela est possible, ARSn met en œuvre des solutions standardisées sur ses machines neuves. Cela permet d'améliorer l'efficacité et la rentabilité des projets, tout en garantissant une qualité optimale.



**Figure 3 : Les Prestations et Activités de ARSn**

ARSn est spécialisée dans le développement de machines spéciales pour répondre aux besoins spécifiques de ses clients. Chaque machine est conçue et fabriquée selon un cahier des charges précis, garantissant une solution parfaitement adaptée à l'application finale. ARSn a choisi de travailler avec les marques leaders en matière d'automatisme et de robotique, telles que Schneider et Siemens. L'entreprise utilise également des robots de différentes marques, telles que STAUBLI, FANUC, OMRON, YASKAWA et ABB, afin de répondre aux exigences spécifiques de ses clients.

ARSn propose des prestations spécifiques dans les domaines suivants :

- **Ligne de montage automobile** : ARSn peut fournir des opérateurs et des robots pour assembler des véhicules sur une ligne de production.
- **Changement et déchargement de machines** : ARSn peut utiliser des robots pour charger et décharger des machines-outils, ce qui permet d'améliorer la sécurité et la productivité.
- **Dévracage** : ARSn peut utiliser des robots pour dévraquer des pièces métalliques, ce qui permet d'éviter les accidents du travail.
- **Soudure par point** : ARSn peut fournir des robots et des outils de soudure par point pour assembler des pièces métalliques.

#### 4.4 Des clients satisfaits



**Figure 4 : Clients satisfaits**

## 4.5 Service

Durant mon alternance en Robotique & Informatique, j'ai eu la chance de rejoindre ARSN, je suis attaché au pôle automobile prestation, j'ai été encadré par FORNEROD Antoine, responsable du pôle, et par plusieurs collègues roboticiens.

Mes missions étaient variées et enrichissantes. J'ai participé à la conception et au développement de solutions robotiques innovantes pour l'industrie automobile. Parmi mes principales réalisations, je peux citer :

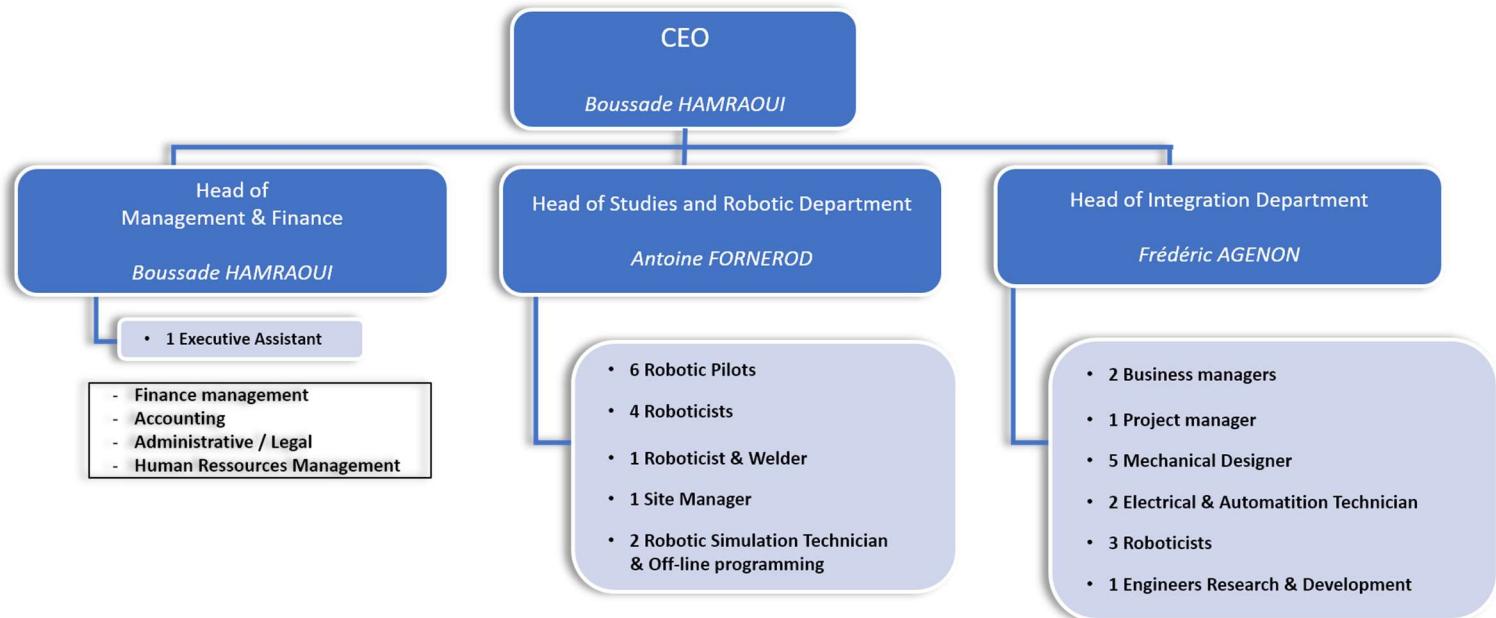
- La simulation de trajectoires robotiques pour optimiser la performance et la sécurité des installations.
- Le développement de programmes robotiques complexes pour des applications de soudure, de montage et de manipulation.

J'ai eu l'opportunité de participer à des missions et déplacement sur site client, pour l'installation et la mise en service des machines. Cette expérience m'a permis de mettre en pratique mes connaissances et de développer mon sens du contact client.

## 4.6 Organigramme

L'organigramme ci-joint présente la structure organisationnelle d'ARSn en date 2023. Il permet de visualiser la répartition des responsabilités et des compétences au sein de l'entreprise, ainsi que les liens hiérarchiques et fonctionnels entre les postes.

Chaque pôle abrite des équipes spécialisées collaborant étroitement pour mener à bien les projets d'ARSn.



*Figure 5 : Organigramme simplifié de l'entreprise*

## 5 LISTE DU PROJET

### Programmation App Web

- Objectif : Déployer un applicatif web destiné à l'industrie du secteur de l'automobile, tout en développant des composants d'interface et de persistance de données

### Programmation App Lourde

- Objectif : Déployer un applicatif bureau exécutable destiné à l'industrie du secteur de l'automobile, tout en développant un procédé d'aiguillage programme robot.

### Simulation et vérification de procédé (Logiciel : RoboGuide) :

- Objectif : Réaliser et simuler les programmes robots d'une installation d'assemblage de porte. Ayant participer au projet robotique. J'ai pu proposer et réaliser l'étude du cycle des systèmes avec les applicatifs. Cette application sert à étudier et démontrer l'utilité de la persistance de données.

### Programmation Robotique (Type : FANUC)

- Objectif : Programmer un robot poly-articulée standard FANUC, procéder aux essais, à la mise au point et à l'installation d'un système d'assemblage de pièces.

## 6 MOTS-CLÉS REPRÉSENTATIFS DU PROJET

- Apprentissage
- Sécurité
- Réseaux
- Web
- Dockerisation
- Robotique Industriel

## 7 PRÉSENTATION DU SYSTÈME

### 7.1 Raisons du projet

Le projet de développement d'une application web de supervision trouve sa justification dans les besoins pratiques et opérationnels rencontrés lors du déploiement de la ligne MEF (Montage des Eléments de Fixation) au sein de l'entreprise ARSn.

Durant le processus de développement et de mise en route de cette ligne, une contrainte visible et mesurable est apparue : en cas de défaut de vissage détecté par le système, le robot n'était pas capable de procéder à une action de resserrage. Cette situation nécessitait soit une réinitialisation complète de l'îlot, soit l'intervention d'un conducteur de ligne, entraînant des arrêts de production non négligeables et impactant la productivité de l'ensemble de la ligne.

De plus, sur les postes de production suivants, les opérateurs n'étaient pas informés en temps réel des vis manquantes, ce qui pouvait entraîner la sortie de portes non vissées sur les voitures, entraînant des erreurs de qualité et potentiellement des risques pour la sécurité.

Dans ce contexte, l'objectif de l'application web de supervision est d'apporter une solution efficiente en remontant toutes les données de la ligne et de la cellule de production. Elle permettra de traiter ces informations en temps réel et d'afficher les actions nécessaires aux opérateurs des postes suivants, par exemple en signalant les défauts de vissage ou en fournissant des instructions pour la correction des anomalies. De plus, cette application offrira la possibilité de suivre les performances de la ligne, de réaliser des statistiques et de mettre en place des actions correctives ou préventives.

Le développement d'une application lourde permettra aux techniciens qualifiés ou aux conducteurs de ligne de prendre le contrôle des robots directement depuis l'interface, facilitant ainsi la résolution rapide des problèmes et l'alimentation de la base de données pour une meilleure analyse des performances et une optimisation continue du processus de production.

### 7.2 Présentation client



Figure 6 : Logo du client

**Stellantis** est l'un des plus grands constructeurs automobiles au monde, formé par la fusion de deux géants de l'industrie automobile : Fiat Chrysler Automobiles (FCA) et le Groupe PSA. Cette fusion a été finalisée en janvier 2021. L'entreprise résultante, Stellantis, possède un portefeuille diversifié de marques automobiles, ce qui en fait l'un des acteurs les plus influents du secteur.

Voici quelques points clés pour mieux comprendre Stellantis :

**Fusion de FCA et du Groupe PSA** : FCA, basé en Italie, était le propriétaire de marques telles que Fiat, Chrysler, Jeep, Alfa Romeo et Maserati. Le Groupe PSA, basé en France, contrôlait des marques comme Peugeot, Citroën, DS Automobiles et Opel/Vauxhall. La fusion de ces deux groupes a créé Stellantis, avec des racines et une présence significative en Europe, en Amérique du Nord et dans d'autres régions du monde.

**Portefeuille de Marques** : Stellantis possède un portefeuille impressionnant de marques automobiles, chacune avec sa propre identité et son histoire. Cela comprend des marques emblématiques comme Fiat, Jeep, Peugeot, Citroën, Opel/Vauxhall, Alfa Romeo, Chrysler et Maserati, ainsi que d'autres marques régionales et spécialisées.

**Présence Internationale** : Stellantis est présent sur les marchés du monde entier. Avec des usines de production, des centres de recherche et développement, des bureaux de vente et des concessionnaires dans de nombreux pays, Stellantis est un acteur majeur de l'industrie automobile mondiale.

**Diversité de Produits** : Le portefeuille de produits de Stellantis couvre une large gamme de segments du marché, allant des voitures compactes aux véhicules utilitaires sportifs (SUV) en passant par les camionnettes et les véhicules utilitaires légers. Cela lui permet de répondre aux besoins et aux préférences des consommateurs dans différentes régions du monde.

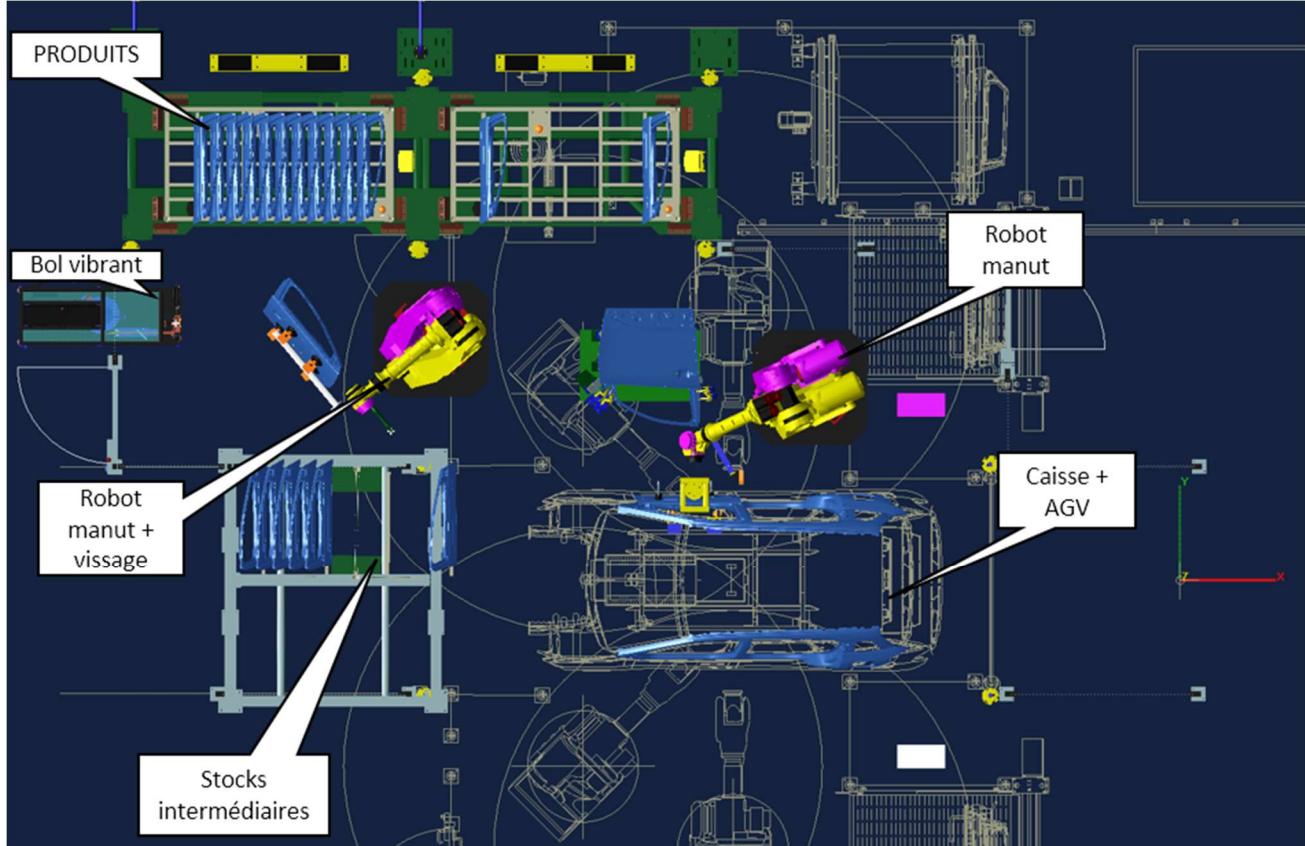
**Engagement envers l'Innovation** : Stellantis est engagé dans le développement de technologies de pointe pour rendre ses véhicules plus sûrs, plus efficaces sur le plan énergétique et plus connectés. Cela inclut des investissements dans des domaines tels que la conduite autonome, l'électrification des véhicules et les services de mobilité intelligente.

En résumé, Stellantis est un acteur majeur de l'industrie automobile mondiale, résultant de la fusion de deux grands groupes, FCA et le Groupe PSA. Avec un portefeuille diversifié de marques, une présence internationale et un engagement envers l'innovation, Stellantis est bien positionné pour continuer à jouer un rôle important dans l'avenir de l'industrie automobile.



Figure 7 : Graphique historique de Stellantis

### 7.3 Détail des fonctionnalités du système actuel



*Figure 8 : Schéma implantation du système MEF-PAV (vue du dessus)*

La ligne de montage MEF Porte avant, comprenant quatre robots poly-articulés. Cette installation est spécifiquement conçue pour réaliser une opération clé dans le processus d'assemblage et de fixation des portes sur les carrosseries de voitures.

La fonctionnalité principale de ce système intégré est d'assembler les portes sur les carrosseries, formant ainsi les véhicules. Deux robots sont chargés de déposer les produits dans des caisses préalablement configurées, exploitant des systèmes de vision pour assurer une précision optimale. Deux autres robots sont affectés à la fixation des portes sur les carrosseries à l'aide de quatre vis de sécurité. L'approvisionnement des caisses s'effectue par des véhicules à guidage automatique (AGV), tandis que les portes sont acheminées en lots de 20 par des caristes. Par ailleurs, un système de bol vibrant est utilisé pour déposer et transporter les vis vers les robots. Ensuite la caisse avec porte est transférée au poste suivant : « montage des ailes avant ». Deux opérateurs sont chargés d'assembler deux pièces phares de la voiture tout en contrôlant le bon vissage des portes précédemment assemblées. Passer cette étape les portes avant ne pourront plus être désassemblées.

## 8 PRÉSENTATION DU PROJET

Dans un premier temps, nous présenterons le sujet dans son ensemble, puis nous détaillerons ses contraintes et les solutions proposées ainsi que les produits qui en sortent. Nous poursuivrons ensuite sur le cahier des charges et sa planification. Nous évoquerons les différentes tâches qui pourront m'être confiées durant cette période avant de dresser un bilan.

### 8.1 Fonctionnalité Générale

La conception et le développement de l'application web de supervision sont motivés par les défis opérationnels rencontrés lors de la mise en place de la ligne MEF (Montage des Eléments de Fixation) chez ARSn.

L'objectif de cette application est de résoudre les problèmes de gestion des défauts de vissage, de supervision des opérations et de communication entre les différents postes de production.

En effet, lors du développement et de la mise en route de la ligne MEF, il est apparu que les robots ne pouvaient pas détecter et corriger les défauts de vissage, nécessitant ainsi une intervention manuelle ou la réinitialisation de tout l'ilot de production. De plus, les opérateurs sur les postes suivants n'étaient pas informés en temps réel des anomalies, ce qui pouvait entraîner des erreurs de qualité et des retards dans la production.

L'application web proposée permettra de remonter en temps réel toutes les données de la ligne de production, de détecter les défauts de vissage et de les signaler aux opérateurs concernés. Elle offrira également des fonctionnalités de supervision et de contrôle pour garantir le bon déroulement des opérations et réduire les temps d'arrêt. De plus, elle permettra de suivre les performances de la ligne, de générer des statistiques et d'optimiser les processus de production.

En résumé, l'application web de supervision répond aux besoins de gestion des défauts de vissage, de supervision des opérations et de communication entre les différents postes de production, contribuant ainsi à améliorer l'efficacité et la qualité de la ligne MEF chez ARSn.

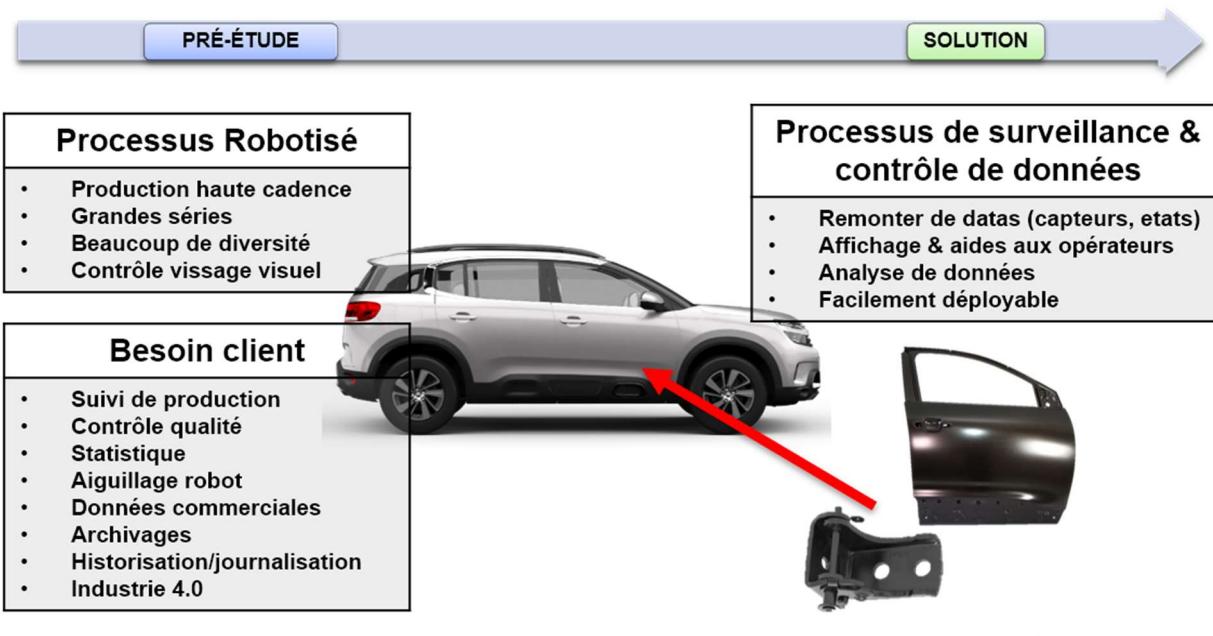


Figure 9 : Visualisation de la transition robotique vers un processus de persistance de données

**Je vais être amené dans le cadre du projet à réaliser 2 applicatifs en autonomie :**

- Pré-étude :
  - ✓ Etablir une solution en concordance avec cahier des charges client
  - ✓ Étudier l'architecture de mon système et de ma solution
  - ✓ Planifier les tâches dans le temps
  - ✓ Définir la méthode de conduite/management de projet
- Analyse :
  - ✓ Choix des composants et solutions à mettre en œuvre
  - ✓ Rechercher les ressources technologiques actuelles
- Conception :
  - ✓ Maquettage des applicatifs
  - ✓ Architecture des données
- Réalisation :
  - ✓ Création d'une Interface Web de supervision (affichage de données)
    - Programmation Backend et frontend
    - Test unitaires
    - Communication inter-outils
    - Dockerisation
    - Sécurités des applicatifs
  - ✓ Création d'une Interface Lourde (requête)
    - Backend et frontend
    - Test unitaires
    - Communication inter-outils
  - ✓ Programmation de Robot Fanuc en langage TPE et KAREL avec Appli background et serveur
- Tests :
  - ✓ Tests des applicatif avant-déploiement
  - ✓ Tests des applicatif post-déploiement
- Déploiement :
  - ✓ Tests des programmes robot
  - ✓ Configurations réseaux.
  - ✓ Tests des Application Web et Lourde simultané
  - ✓ Crédit d'une vidéo technique de démonstration
- Couplage système à l'installation :
  - ✓ Réception de la machine par le client
  - ✓ Installation sur site de l'usine

## 9 FLANIFICATION & GESTIONS DES TACHES

### 9.1 GANTT Simplifié

Le GANTT représente graphiquement les différentes tâches de mon projet sur une ligne de temps entre Septembre et Avril. Pour le développement d'une application légère ou lourde, le diagramme de Gantt est essentiel car il me permet de planifier les différentes étapes du projet, d'assigner des ressources aux tâches, de définir des dépendances entre les différentes activités, et de suivre l'avancement global du projet. Cela nous aide à maintenir le projet pour respecter les délais et les objectifs fixés.

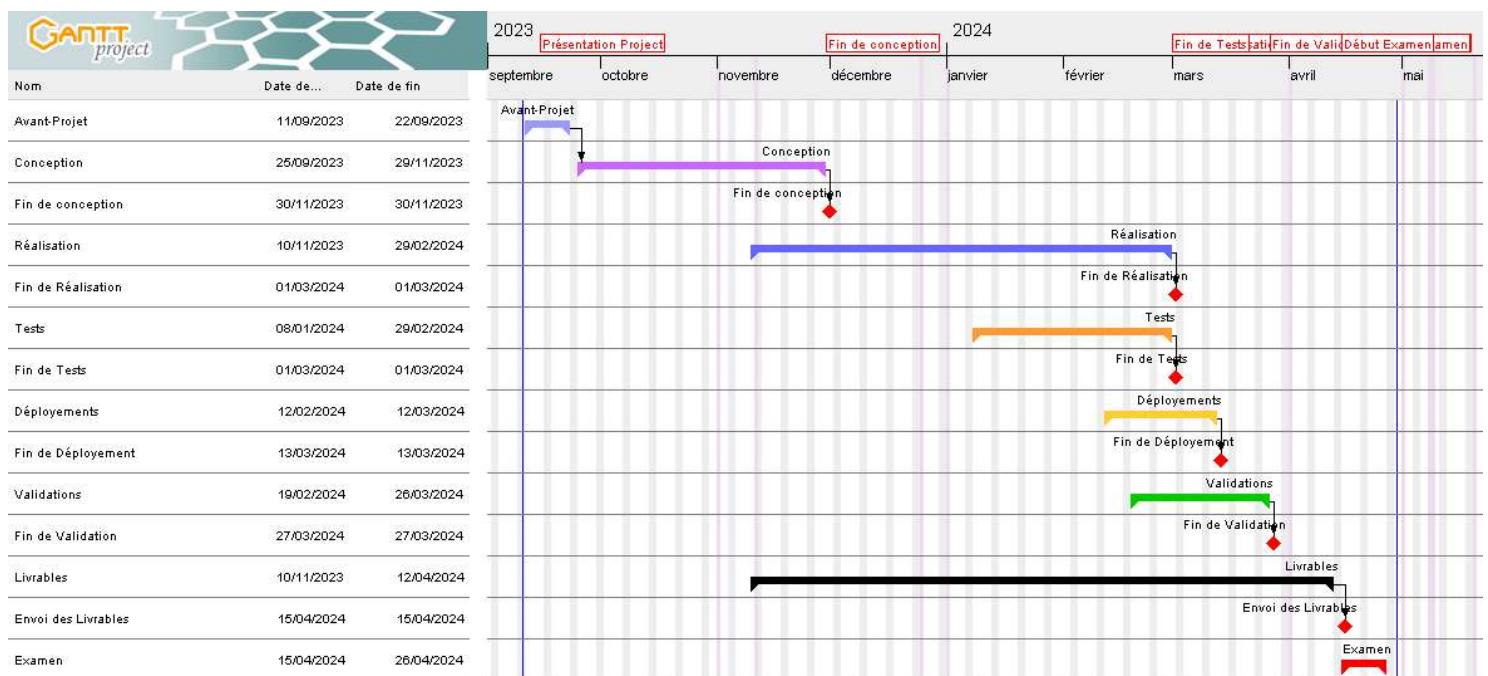


Figure 10 : GANTTS & Planifications des tâches

Voir Annexes GANTT pour plus de précision (développement, des taches)

### 9.2 Méthode Agile

L'adoption de la méthode agile pour gérer mon projet m'a aidé à maintenir une vision claire tout au long du développement. En visualisant facilement les tâches accomplies et celles qui restent à accomplir, j'ai pu mieux organiser mon temps de travail et optimiser ma productivité.

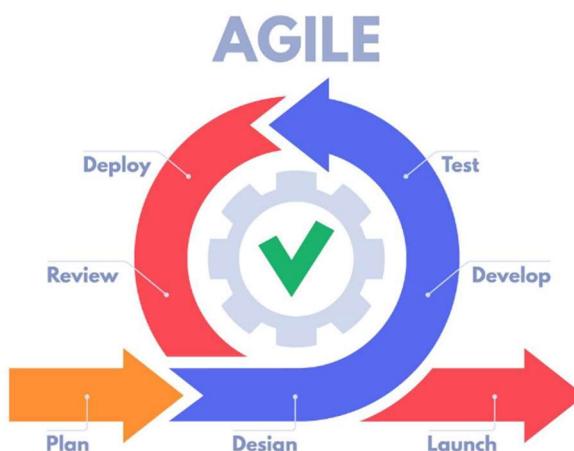


Figure 11 : Illustration management Agile

## 9.3 Méthode versionnage projet

GitHub est un site de partage de code, sur lequel on peut publier des projets dont le code est géré avec le système de gestion de version Git. Structurer son dépôt GitHub en plusieurs fichiers garanti une organisation efficace de son projet. Cette approche permet une meilleure lisibilité et maintenabilité du code, ce qui est essentiel pour assurer la pérennité du projet.

📁 ARCHIVES	MAJ fonctionnel fin proj...	3 weeks ago
📁 CDA_ROBOT	MAJ fonctionnel fin proj...	3 weeks ago
📁 DEPLOIEMENT	MAJ Passage en V2.0.0 v...	3 weeks ago
📁 LOURD	Commanditaire	yesterday
📁 MAQUETTE	MAJ readme notice instr...	last month
📁 RESSOURCES	Create stacktest.yml	last month
📁 WEB	locust maj	6 hours ago
📄 README.md	MAJ Readme	last month

Figure 11 : Structure du repository Github

Les branches dans un dépôt GitHub sont des copies isolées du code principal elles permettent de travailler sur des fonctionnalités, des correctifs ou des améliorations sans perturber le code en production.

**Branch Modification** : Cette branche est destinée à contenir les modifications et les ajouts de fonctionnalités

**Branch Déploiement** : Une fois que nos modifications sont prêtes à être testées et déployées

**Branch Production** : Une fois que nos modifications ont été testées avec succès sur la branche de déploiement, cette branche contient le code qui est déployé en production et accessible aux utilisateurs finaux.

Branch	Updated	Check status	Behind	Ahead
Modification	6 hours ago			Default
Deploiement	3 weeks ago		5	2
Production	6 months ago		250	0

Figure 12 : Branches Github

Les Tags permettent de référencer les versions spécifiques de notre code. Créer un tag pour chaque version majeure de notre logiciel permet aux utilisateurs de facilement télécharger et utiliser des versions spécifiques du logiciel.

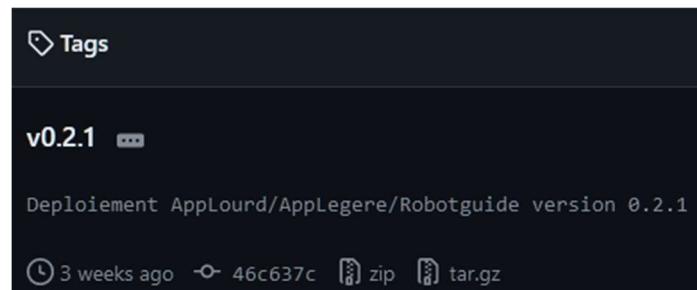


Figure 13 : Exemple d'un versionnage Github

## 10 FONCTIONS DÉTAILLÉES

### 1. AGV

Véhicule automne apportant la caisse tôle dans l'installation automatisée & robotisée. Guidé par une marque au sol et des « tags » électronique, il est complètement autonome et connais sa position exacte dans la ligne de production. Il interagi avec son environnement grâce à son laser et radar de sécurité.



Figure 14 : Vue AGV

### 2. Dévracage Vis

Les vis seront chargées manuellement dans 1 trémie dédiée permettant plusieurs heures d'autonomie qui permet de rendre disponible chaque vis unitairement et orienter correctement à la sortie du bol.

### 3. Robot Vissage

Le robot est un robot **FANUC 6 axes R-2000iB/165F** avec sa baie de contrôle R-30iB. Il est équipé d'une main de préhension à plusieurs ventouses indépendantes ainsi que plusieurs taquets, on retrouve également une visseuse Dessouter, permettant de manipuler les portes et les vis. Ce robot prend une porte dans les Clayes (conteneur produit), la place dans un poste de re taquage puis retrouve prendre une porte à dépose sur un poste tampon, enfin il prend une vis avant de se diriger vers la caisse. Il doit visser 4 vis de sécurité une à une.



Figure 15 : Vue robot

### 4. Poste re taquage

Ce sous-ensemble est un poste d'indexage. Le robot vient donc charger les portes tour à tour. Des capteurs de présence permettent de contrôler l'état du chargement et le bon référencement de la pièce.

### 5. Poste tampon

Ce poste est un système permettant le stockage temporaire de portes

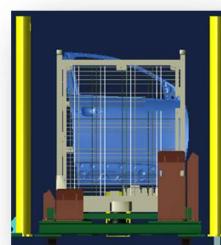


Figure 16 : Vue Tampon

### 6. Robot Vision

Le robot est un robot **FANUC 6 axes R-2000iB/165F** avec sa baie de contrôle R-30iB. Il est équipé d'une main de préhension à plusieurs ventouses indépendantes ainsi que plusieurs taquets, on retrouve également une caméra vision, permettant de manipuler les portes et orienté le produit par rapport à la caisse. Ce robot prend une porte dans le poste de taquage, la place dans la voiture. Il doit attendre que le robot de vissage ait terminé son travail.

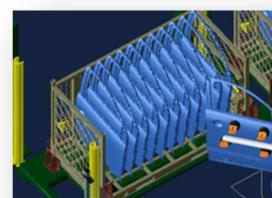


Figure 17 : Vue Claye

### 7. Claye d'approvisionnement Porte

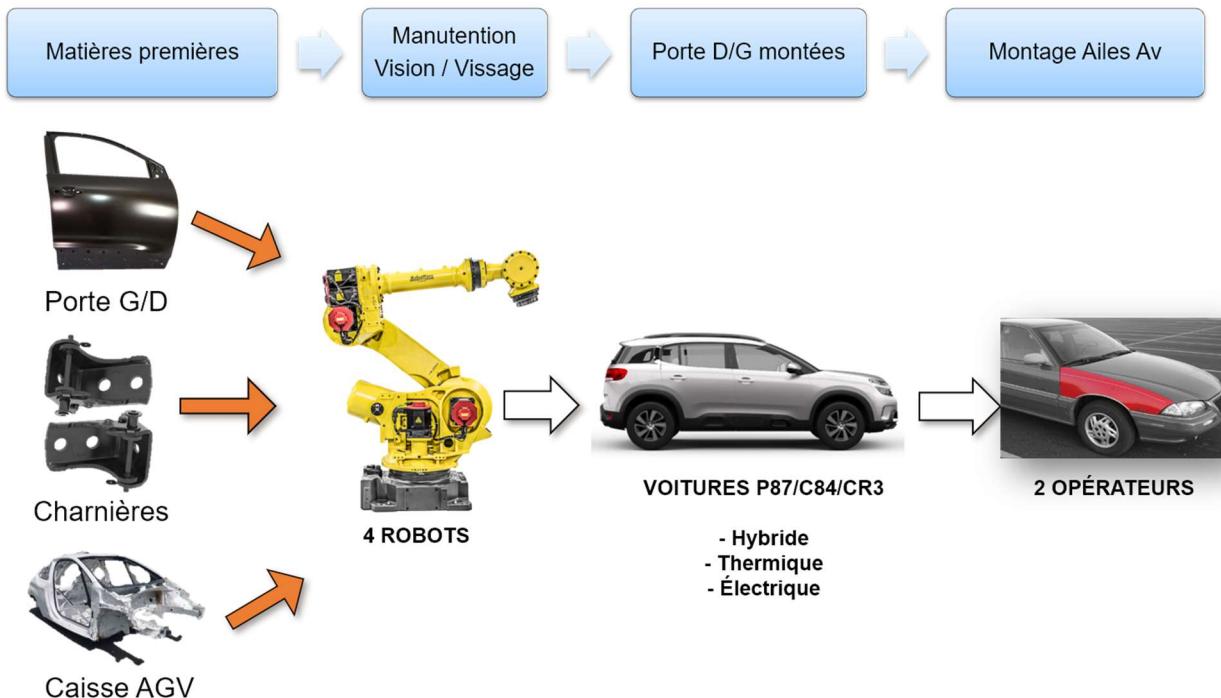
L'installation comprend 4 conteneurs permettant de recevoir 10 portes chacun. Ces racks sont équipés d'avaloirs pour la porte. Un système de bridage permet de maintenir les portes dans le conteneur

## 11 CYCLE SIMPLIFIÉ

Diagramme de cycle système permettant de comprendre le cycle du robot jusqu'au revissage avec les sous-ensembles qui l'entourent.

*Voir Annexes pour voir le diagramme simplifié du cycle.*

## 12 LIGNE ILLUSTRÉE



*Figure 18 : Schéma ligne montage de porte MEF*

Élément clés et légende de schéma :

- Les étapes du processus : Elles sont représentées par des formes géométriques rectangulaires
- Le flux de travail : Il est représenté par des flèches qui relient les différentes étapes du processus.

Avantages d'utiliser un schéma simplifié et imagé du flux et composants :

- Facile à comprendre : Les images et les symboles permettent de visualiser le processus de manière intuitive.
- Concis : Le schéma ne présente que les informations essentielles.
- Flexible : Le schéma peut être facilement modifié pour refléter les changements du processus.

## 13 PRESENTATION DES PRODUITS

### 13.1 Pièces Primaires

Réf	Description	Actions
01	<ul style="list-style-type: none"> <li>Charnières</li> </ul> 	<p>3 références de charnières :</p> <ul style="list-style-type: none"> <li>Matériaux : Acier</li> <li>Jointure entre « Porte » &amp; « Caisse »</li> </ul>
02	<ul style="list-style-type: none"> <li>Porte avant G &amp; D</li> </ul> 	<p>3 références de portes :</p> <ul style="list-style-type: none"> <li>Matériaux : Acier</li> <li>Positionnement dans caisse Gauche &amp; Droite</li> </ul>
03	<ul style="list-style-type: none"> <li>Caisse tôlée</li> </ul> 	<p>3 références caisse :</p> <ul style="list-style-type: none"> <li>Matériaux : Acier</li> <li>Reçois Porte et charnière puis action de vissage</li> </ul>

Figure 19 : Tableau des pièces primaires

### 13.2 Pièce Finie

Réf	Description	Illustration
01	<ul style="list-style-type: none"> <li>Caisse vissée avec porte montée</li> </ul>	

Figure 20 : Tableau de la pièce finie

## 14 DESCRIPTIF DE LA LIGNE

### 14.1 Fonctionnalités de l'installation actuelle (Robotisation simple)

L'îlot robotisé a les fonctions principales suivantes :

- Manutentions d'1 pièces Porte depuis Claye, puis Vis du bol vibrant vers la presse caisse
- Dépose de la porte pour l'assemblage, conformisation, puis vissage des 4 vis de sécurité

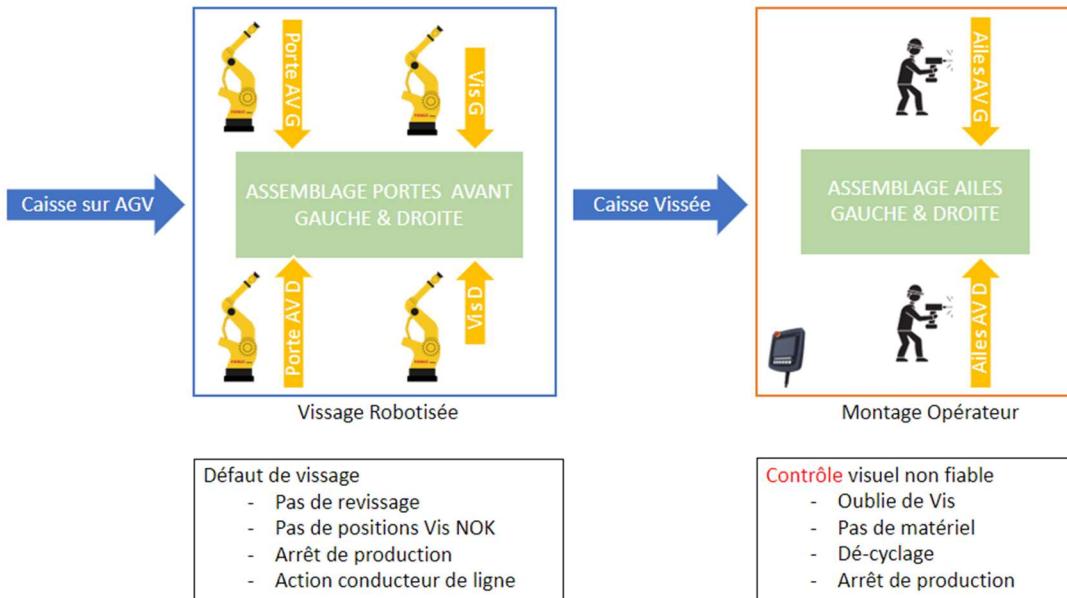


Figure 21 : Schéma fonctionnalités actuelle

### 14.2 Fonctionnalités de l'installation proposé (Persistance de données)

L'îlot robotisé aura les fonctions ajoutées suivantes :

- Collecte de données pour la production de voiture
- Statistique à but commercial

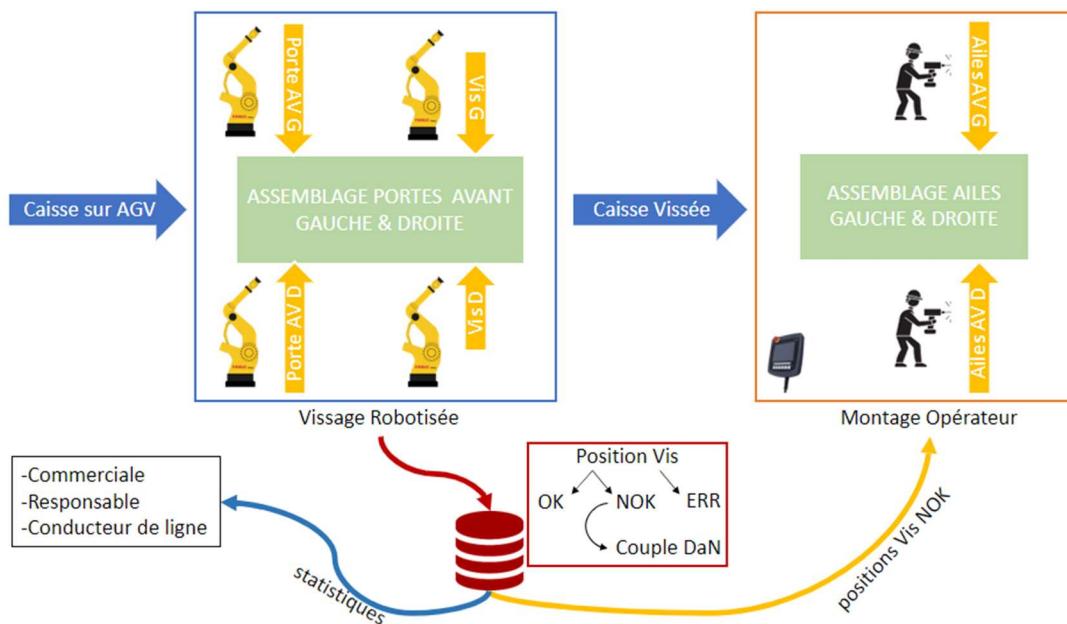


Figure 22 : Schéma fonctionnalités proposées

## 15 ÉCHANGES & DONNÉES

### 15.1 Échanges entre les différents clients

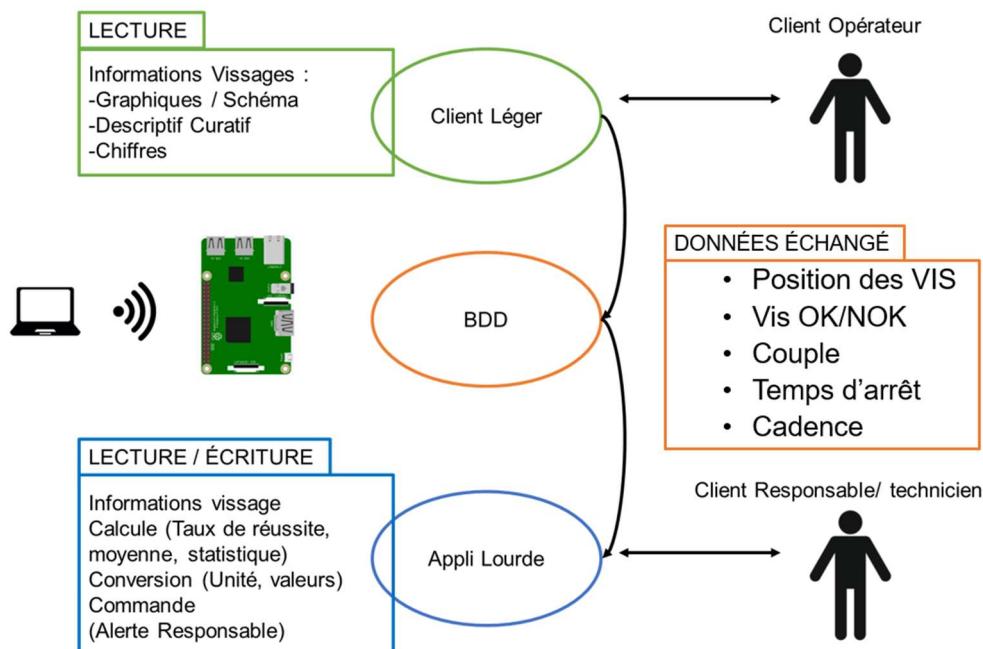


Figure 23 : Schéma simplifié des échanges de données

- **Client Léger (Opérateur de Production) :**  
Représenté par une interface web conviviale accessible depuis un navigateur, le client léger permet à l'opérateur de production d'accéder facilement à des informations cruciales sur la production en cours. Cela inclut les données de capteurs en temps réel, les graphiques illustrant les objectifs commerciaux et les alertes préventives.
- **Base de Données (Hébergée sur un PC : Raspberry Pi, Windows, Linux) :**  
La base de données, symbolisée par un serveur centralisé, joue un rôle essentiel dans le stockage et la gestion des données liées au processus de vissage. Elle héberge des informations telles que la position des vis, l'état de validation, le couple, le temps d'arrêt et la cadence. La technologie de *Dockerisation* est utilisée pour garantir la portabilité et l'efficacité du déploiement des applications.
- **Application Lourde (Destinée au Client Responsable/Technicien) :**  
L'application lourde est représentée par un logiciel installé sur le poste de travail du client responsable ou du technicien. Cette application offre des fonctionnalités avancées telles que la création et la gestion d'ordres de vissage, la surveillance du couple de serrage des visseuses, ainsi que la conversion et le calcul de statistiques pour évaluer les performances du processus de production.

Ensemble, ces composants forment un système intégré qui permet une gestion efficace et optimisée du processus de revissage, offrant des interfaces adaptées aux différents niveaux d'utilisateur, de l'opérateur de production au responsable ou technicien en charge de la supervision et de l'optimisation du processus.

## 15.2 Diagramme de pieuvre

Gestion des utilisateurs : Cette partie du diagramme représente les fonctionnalités liées à la gestion des utilisateurs de l'application, la connexion, la gestion des profils utilisateurs, la réinitialisation de mot de passe.

Collectes des données : Cette section met en évidence les fonctionnalités permettant aux utilisateurs de visualiser et d'analyser les données pertinentes. Les graphiques interactifs, les tableaux de bord personnalisables, etc.

Gestion des contenus : Cette partie du diagramme représente les fonctionnalités permettant aux utilisateurs de gérer le contenu de l'application, tels que la création, la modification et la suppression de données, le téléchargement de fichiers, la publication de contenu, etc.

Communication et collaboration : Cette section du diagramme met en évidence les fonctionnalités facilitant la communication et la collaboration entre les utilisateurs, telles que les forums de discussion, les messageries instantanées, les commentaires sur les publications, etc.

Gestion des autorisations : Cette partie du diagramme représente les fonctionnalités permettant de définir et de gérer les autorisations d'accès des utilisateurs aux différentes parties de l'application. Cela pourrait inclure la définition de rôles et de permissions, la gestion des groupes d'utilisateurs, etc.

Intégrations avec d'autres systèmes : Cette section du diagramme met en évidence les fonctionnalités permettant à l'application de s'intégrer avec d'autres systèmes ou services externes, tels que les API tierces, les importations/exportations de données, les connexions avec des plateformes externes, etc.

Sécurité : Cette partie du diagramme représente les fonctionnalités visant à garantir la sécurité des données et des utilisateurs de l'application, telles que l'authentification à deux facteurs, le cryptage des données, la protection contre les attaques par injection SQL, etc.

En expliquant votre diagramme de pieuvre, assurez-vous de mettre en évidence comment chaque fonctionnalité contribue à l'objectif global de l'application et comment elles interagissent entre elles pour offrir une expérience utilisateur cohérente et efficace.

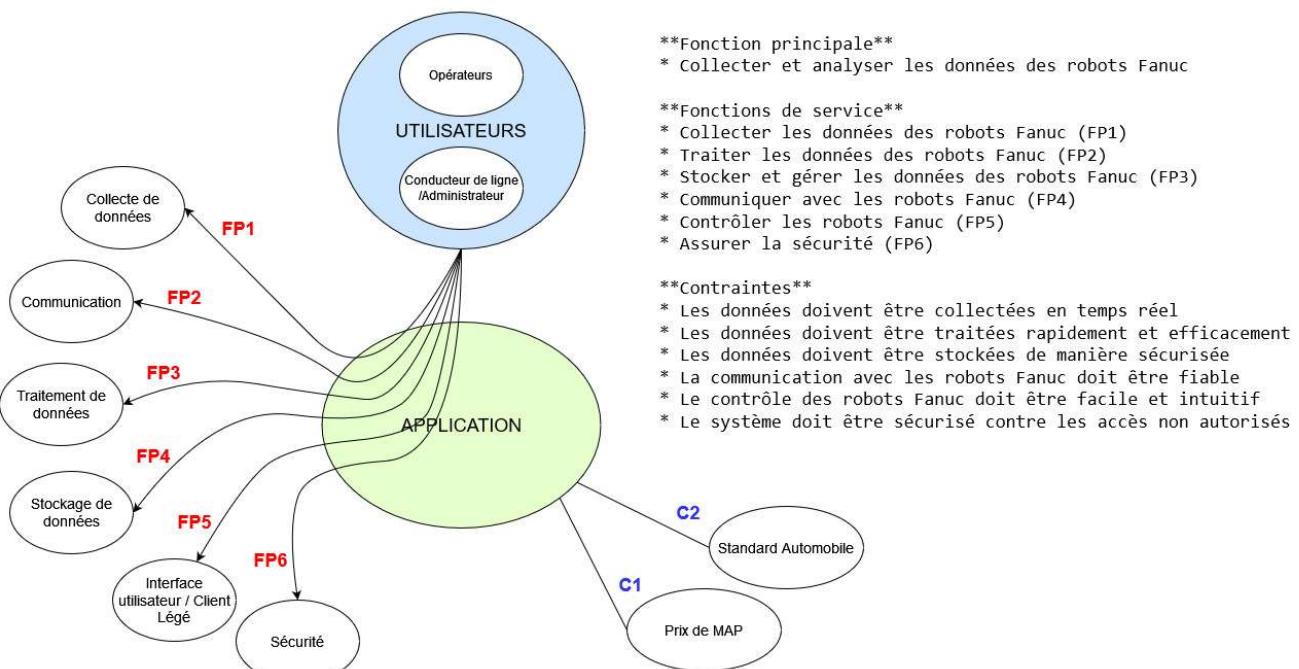


Figure 24 : Diagramme de pieuvre

## 15.3 Fonctions Principales

FONCTIONS		DESCRIPTION
PRINCIPALES FONCTIONS		
FP1	Collecte de Données	
FP1.0	Collecte des données des robots Fanuc	
FP1.1	Traitement initial des données brutes	
FP1.1	Validation et filtrage des données collectées	
FP2	Traitement des Données	
FP2.0	Conversion des données dans le format requis	
FP2.1	Calcul de statistiques (moyennes, écarts-types, etc.)	
FP2.2	Génération de rapports de données	
FP2.3	Contrôle qualité des données (détection des données erronées ou incohérentes)	
FP3	Stockage et Gestion de Données	
FP3.0	Stockage des données dans la base de données SQL (MySQL)	
FP3.1	Gestion des autorisations d'accès aux données	
FP3.2	Sauvegarde et récupération de données en cas de panne	
FP4	Communication	
FP4.0	Communication entre les robots Fanuc et le Client Lourd	
FP4.1	Transfert des données traitées Client Lourd au serveur SQL	
FP4.2	Communication entre Client léger et le serveur SQL	
FP5	Interface Utilisateur Client Léger	
FP5.0	Contrôle des robots Fanuc	
FP5.1	Surveillance en temps réel de l'état des robots	
FP5.2	Notifications en cas de problèmes ou d'alarmes	
FP6	Sécurité	
FP6.0	Authentification des utilisateurs	
FP6.1	Mesures de sécurité pour protéger l'accès au système	

**Figure 25 : Tableau des fonctions principales**

Cette figure décrit les principales fonctions de notre système de gestion du processus de vissage.

- FP1. Collecte de Données :** Cette fonction implique la collecte des données des robots Fanuc et leur traitement initial, y compris la validation et le filtrage pour garantir leur qualité.
- FP2. Traitement des Données :** Une fois collectées, les données sont converties dans le format requis, puis des statistiques sont calculées. Des rapports sont également générés, et un contrôle qualité est effectué pour détecter les données incorrectes.
- FP3. Stockage et Gestion de Données :** Les données sont stockées dans une base de données SQL (MySQL), avec des mécanismes de gestion des autorisations et de sauvegarde en cas de panne.
- FP4. Communication :** Les données sont échangées entre les robots Fanuc et le Client Lourd, puis transférées vers le serveur SQL. Le Client Léger communique également avec le serveur SQL.
- FP5. Interface Utilisateur Client Léger :** Cette fonctionnalité permet le contrôle des robots, la surveillance en temps réel de leur état.
- FP6. Sécurité :** Des mesures d'authentification des utilisateurs et de sécurité sont mises en place pour protéger l'accès au système.

## 15.4 Allocation

ALLOCATION N+1		
FONCTIONS	HW	SW
FP1	X	X
FP2		X
FP3		X
FP4	X	X
FP5	X	X
FP6		X

ALLOCATION N+2					
FONCTIONS	HW: Robot	HW: PC	SW : Lourd	SW: Legé	SW: Robot
FP1	X	X	X		X
FP2		X	X	X	X
FP3	X	X	X	X	
FP4	X	X	X	X	
FP5			X	X	
FP6	X	X	X	X	X

Figure 26 : Tableau allocations des fonctions

L'allocation N+1 combine matériel et logiciel pour les principales fonctions telles que la collecte, le traitement et la communication des données, tandis que la sécurité est entièrement gérée par le logiciel.

En revanche, l'allocation N+2 répartit les fonctions entre différents matériels, notamment les robots et les PC, et entre plusieurs types de logiciels, tels que les applications lourdes, légères et celles dédiées aux robots, pour une exécution plus spécialisée et optimisée. Cette répartition permet une utilisation efficace des ressources matérielles et une adaptation aux besoins spécifiques de chaque composant et utilisation du système.

## 16 RÔLES & FONCTIONNALITÉES

### 16.1 Operateur

Les rôles opérateurs peuvent accéder aux fonctionnalités de base requis pour la production. L'opérateur à un droit de lecture, il ne peut prendre de décisions sur le fonctionnement de l'installation. Il n'a accès qu'à l'applicatif web, installé à son poste.

### 16.2 Conducteur de ligne

Les rôles conducteurs d'installations (CI) peuvent accéder aux fonctionnalités de base et avancé requis pour la production. Le CI à un droit d'édition et de décision sur le process de l'installation. Il participe directement à l'aiguillage des robots. Il utilise principalement l'applicatif lourd.

### 16.3 Invité / Commercial

Les rôles invités ou commerciaux sont limités à une seule fonctionnalité. Ils utilisent la page statistique de l'applicatif web pour interagir avec les client et fournisseurs.

### 16.4 Administrateur

Les rôles administrateurs peuvent accéder à toutes les fonctionnalités d'un utilisateur enregistré, cependant ceux-ci possèdent également un accès à certaines parties de l'administration comme la création et modification de mot de passe.

M4-CDA (Concepteur développeur d'applications)

## 17 CAS D'UTILISATION DES APPLICATIFS

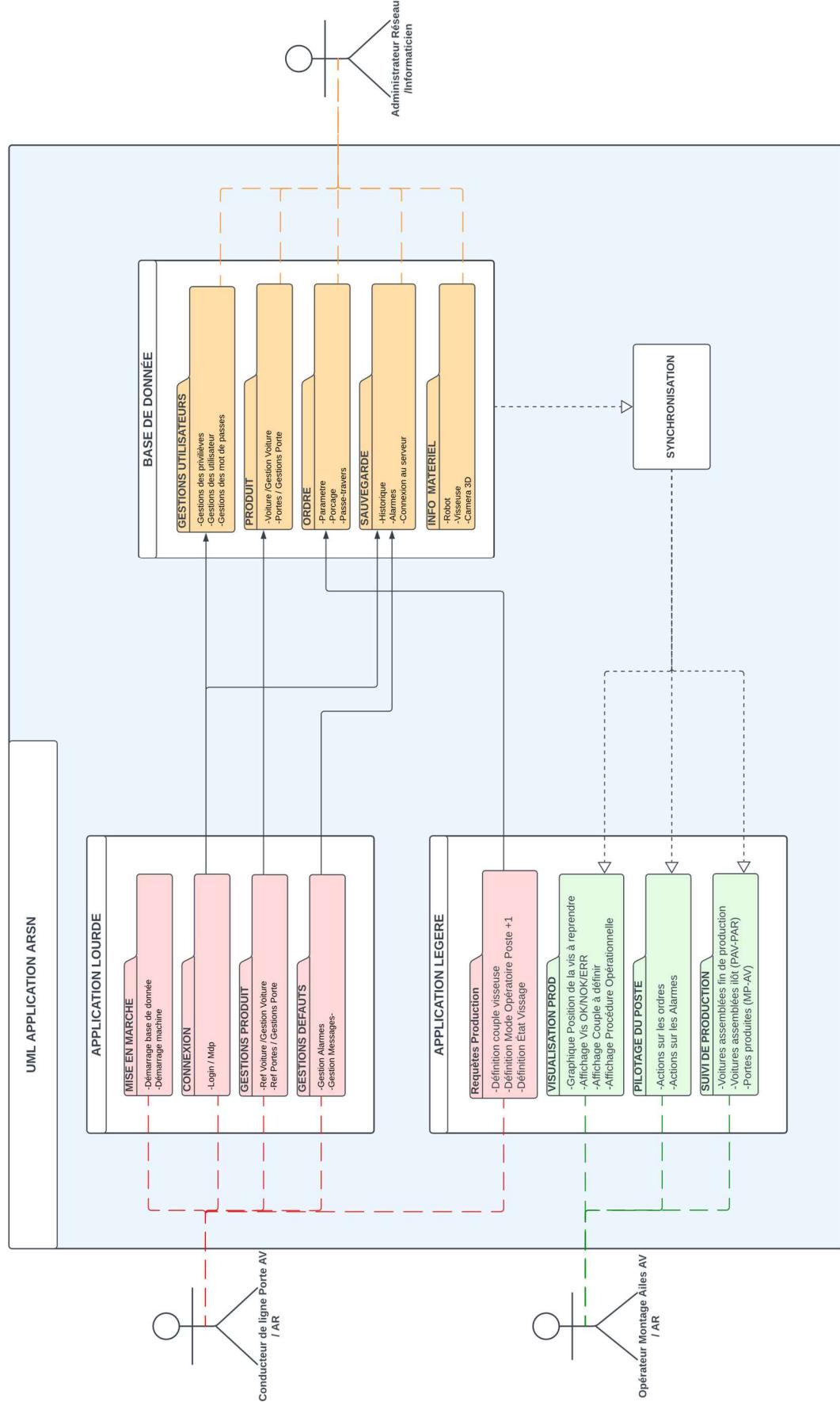


Figure 27 : Figure cas d'utilisation des applicatifs

M4-CDA (Concepteur développeur d'applications)

## 18 DICTIONNAIRE DE CLASSES / DONNÉES

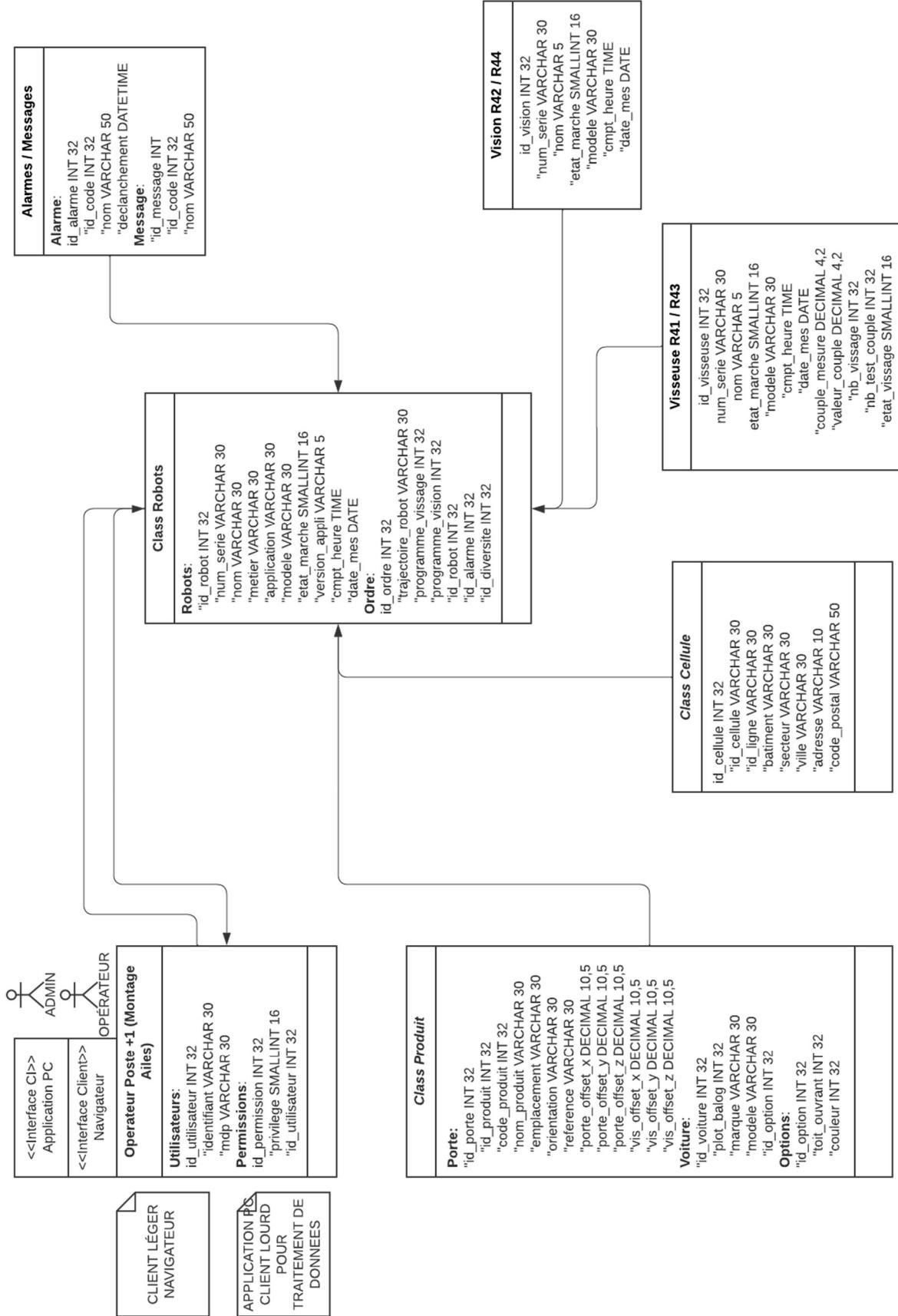


Figure 28 : Figure dictionnaire de classes & données

## 19 MODÉLISATION DE LA BDD

### 19.1 Méthode MCD

#### 19.1.1 Processus d'attaque

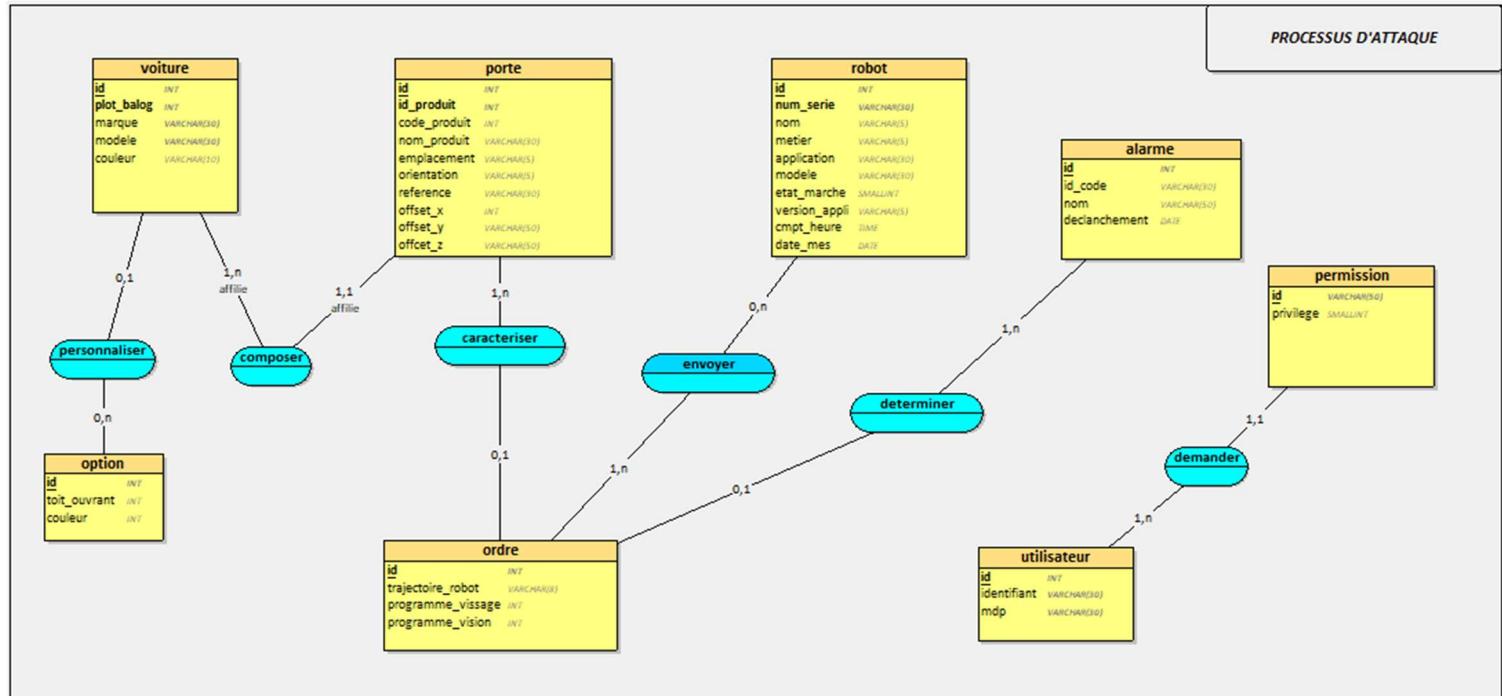


Figure 29 : Représentation des données du système du processus de fonctionnement

#### 19.1.2 Processus d'affichage

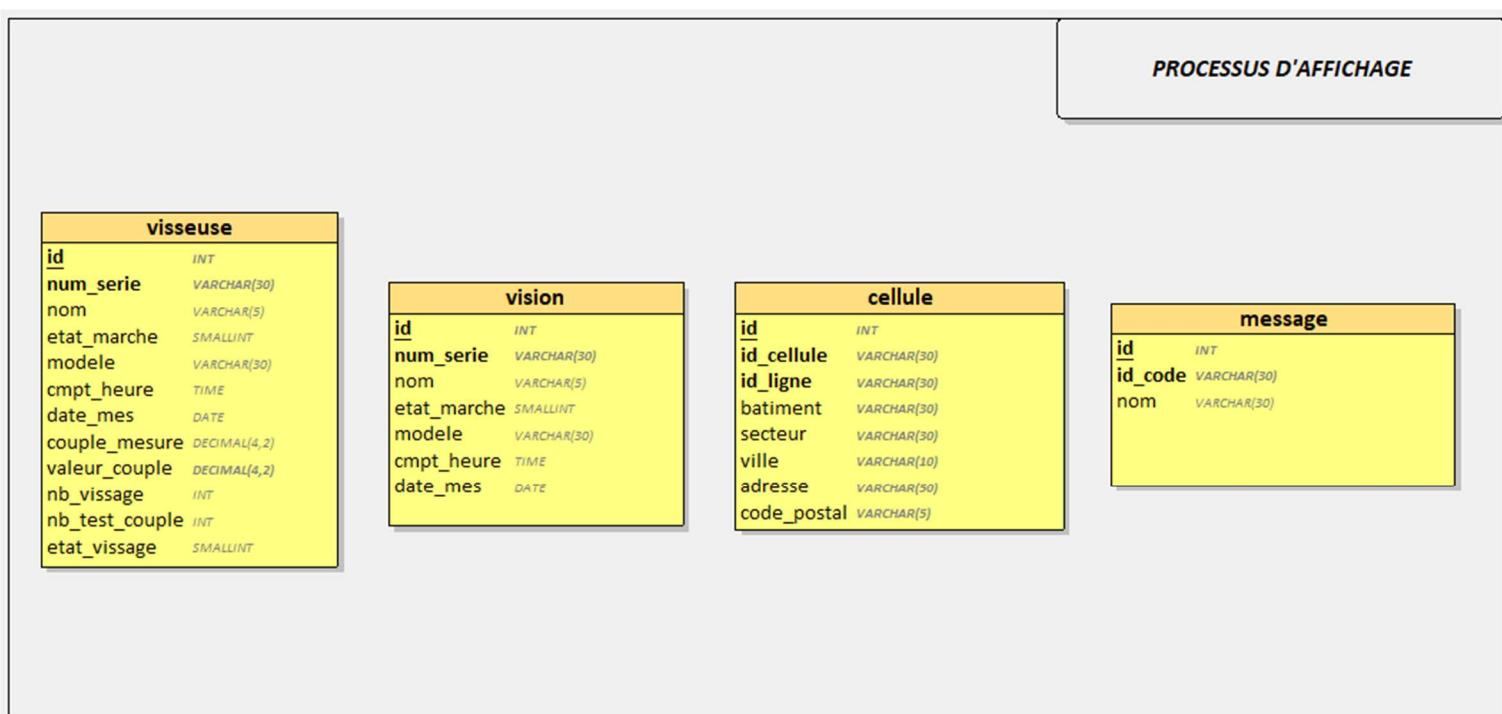


Figure 30 : Représentation des données du système affichage

## 19.2 Modélisation de la BDD méthode MLD

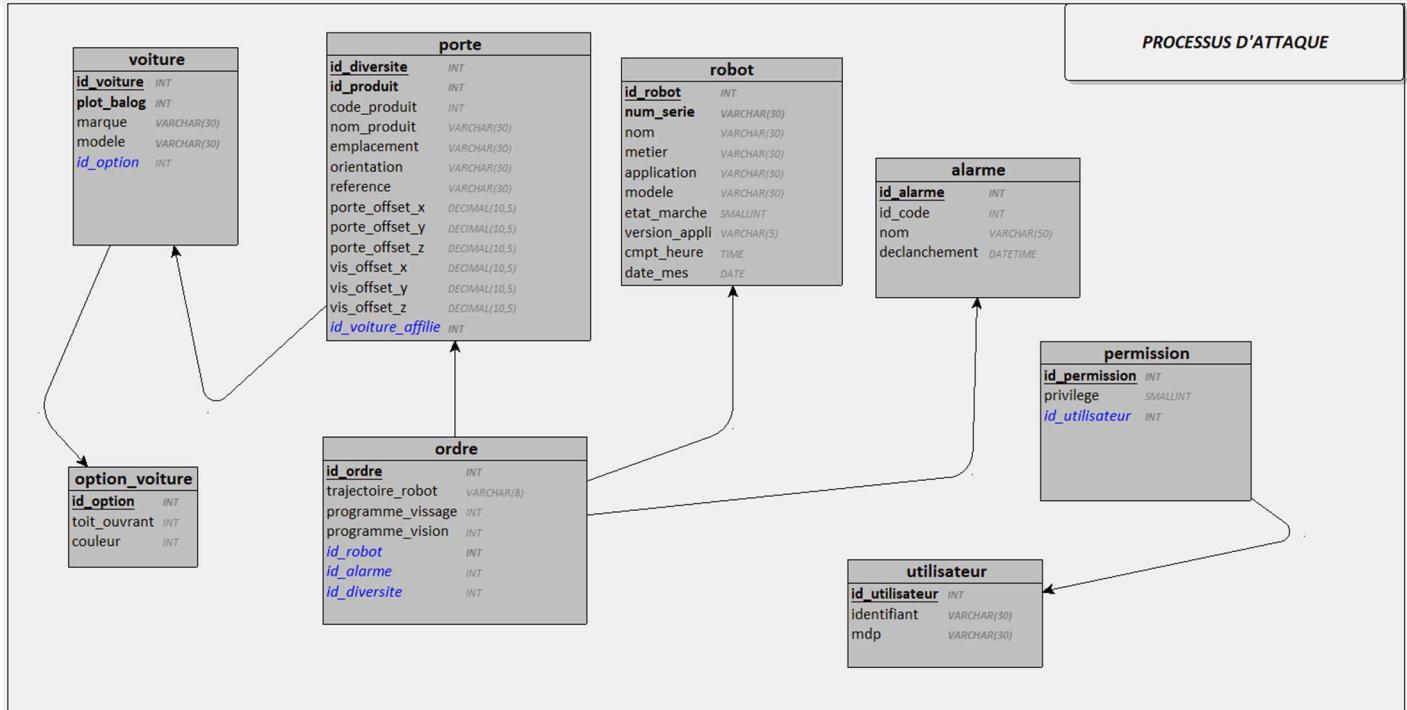


Figure 31 : Représentation méthode MLD avec clés étrangères

Les cardinalités sont des caractères (**0,1, n**) qui fonctionnent par couple et qui sont présents de chaque côté d'une association. Elles donnent des indications très intéressantes et permettent par la suite de construire la base de données :

- Avec la création de clés étrangères dans le cas d'une CIF (**Contrainte d'Intégrité Fonctionnelle**)
- Avec la création d'une table intermédiaire dans le cas d'une CIM (**Contrainte d'Intégrité Multiple**)

Les cardinalités possibles sont :

- 0,1 : au minimum 0, au maximum 1 seule valeur (CIF)
- 1,1 : au minimum 1, au maximum 1 seule valeur (CIF)
- 0, n : au minimum 0, au maximum plusieurs valeurs
- 1, n : au minimum 1, au maximum plusieurs valeurs

Par exemple : dans l'image « Processus d'attaque MLD », ci-dessus, on a « 1,1 » en cardinalité du côté « ordre » de l'association et une cardinalité « 0, n » du côté « robot ». Concrètement, il faut lire les cardinalités ainsi :

- Un ordre est envoyé à 1 et 1 seule robot
- Un robot peut se recevoir 0 ou plusieurs (n) ordres.

## 20 MAQUETTAGE WEB



Figure 32 : Prototypage & maquettage des pages web

Le maquettage permet au client de visualiser le design et les fonctionnalités de l'application. Cela facilite la communication et la validation des attentes dès les premières étapes du projet.

**Validation et Rétroaction :** En présentant des maquettes interactives, nous recueillons les retours du client sur le design, la disposition des éléments et les fonctionnalités proposées. Cela permet d'ajuster rapidement les éléments qui ne correspondent pas aux attentes, réduisant ainsi les erreurs et les coûts de correction ultérieurs.

**Guidage pour le Développement :** Les maquettes servent de guide visuel clair pour les développeurs, leur fournissant des indications précises sur ce qui doit être implémenté et comment. Cela réduit les ambiguïtés et les malentendus, assurant une cohérence entre la vision du client et la réalisation technique.

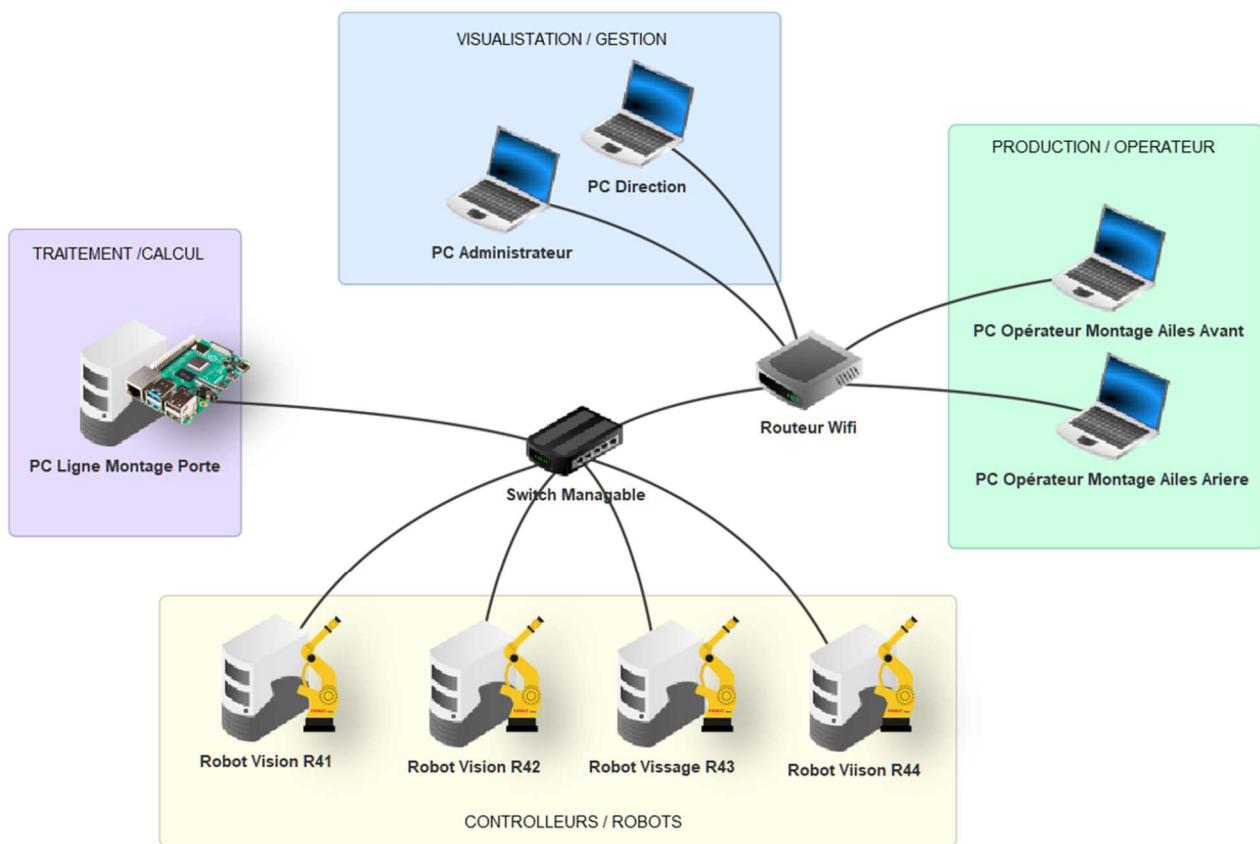
**Gestion de Projet :** Les maquettes permettent une planification efficace du projet en définissant les différentes pages, fonctionnalités et interactions prévues. Elles servent également de référence pour la gestion des ressources et des tâches, facilitant ainsi la coordination et le suivi du projet.

Le maquettage Figma a été crucial pour aligner les attentes du client, valider le design et les fonctionnalités, guider le développement et faciliter la gestion de projet, contribuant ainsi au succès global de l'application web.

## 21 ARCHITECTURE MATERIEL

### 21.1 Architecture réseau du système

L'illustration met en avant l'îlot de conformisation et de vissage des portes avant gauche et droite. Nous utilisons le protocole **TCP/IP** pour garantir la connectivité et la communication entre les divers composants du réseau, bénéficiant ainsi de sa fiabilité, stabilité et de sa compatibilité avec une variété d'appareils et de systèmes.



**Figure 33 : Illustration de l'architecture matériel de la solution proposé**

**Postes de Travail :** Ils disposent de terminaux de contrôle et de supervision, connectés au réseau principal.  
**Robots :** Ces robots, intégrés à l'îlot de conformisation et de vissage des portes, sont contrôlés à distance via des interfaces *Ethernet*.

**Serveur de Supervision :** Centralisé, il collecte et analyse les données pour surveiller les performances de la ligne de montage.

**Switch Réseau :** Assurant la connectivité entre les différents appareils et systèmes du réseau.  
 En utilisant le protocole **TCP/IP** et en attribuant des *adresses IP* cohérentes, nous avons établi une base robuste pour assurer la connectivité et la communication entre les composants du réseau, mettant ainsi en lumière les principaux composants et leur interconnexion.

## 21.2 Architecture réseau du système

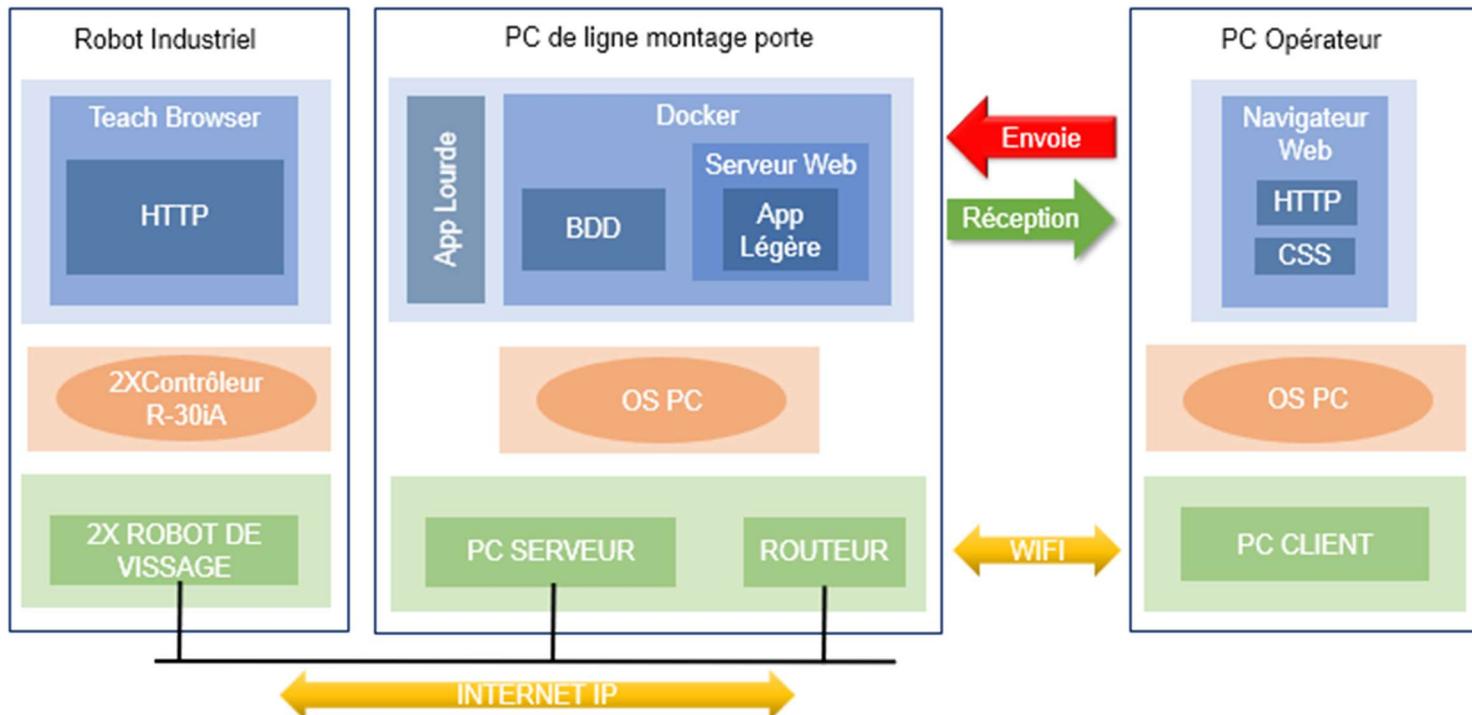


Figure 34 : Schéma des 3 grandes couches réseau et interaction matériel

Notre application web pour l'industrie est composée de trois grandes couches distinctes :

### Couche1. Matériel :

- *Robots industriels* : Utilisés pour effectuer les tâches, comme le vissage.
- *PC de ligne* : Utilisé pour contrôler les robots de vissage & surveiller le processus de production.
- *PC opérateur* : Ils sont utilisés par les opérateurs pour interagir avec l'application web et suivre la production.

### Couche2. Les systèmes d'exploitation (OS) :

- *OS PC* : Système d'exploitation (Windows & Linux) des PC de ligne et des PC opérateur.
- *OS robot* : Il s'agit du système d'exploitation des robots industriels.

### Couche3. Les logiciels :

- *Docker* : Plateforme de conteneurisation qui permet de déployer et de gérer des applications.
- *Navigateurs internet* : Utilisés pour accéder à l'application web.
- *Serveur web* : Il s'agit du serveur qui héberge l'application web (hébergement sur le local).

**Interaction :** Les robots industriels sont connectés aux PC de ligne de montage. Les PC de ligne de montage sont connectés au serveur web. Les PC opérateur sont connectés au serveur web.

L'application Lourde envoie des instructions aux robots industriels via les PC de ligne de montage. Les opérateurs peuvent utiliser l'application web pour surveiller la production.

**Voir Annexes Couches et protocoles pour plus de précision (architecture de développement et déploiement)**

## 21.3 Choix du matériel

Au niveau du choix du matériel, si le client n'exige aucune marque ou aucun équipement spécifique, nous essayons de garder du matériel standard. L'objectif étant encore une fois de pouvoir déployer rapidement notre solution et de gagner un maximum de temps au déploiement et la mise en route.

Nous avons opté pour le *Raspberry Pi* (micro-ordinateur) en raison de sa taille compacte, de sa faible consommation d'énergie et de sa polyvalence, ce qui en fait une solution idéale pour les applications embarquées dans le domaine de l'automobile. Le PC Windows a été choisi pour le développement de l'application lourde en raison de sa compatibilité avec de nombreux logiciels de simulation robotique (*Roboguide*).

Nous avons utilisé une machine virtuelle *Linux* via *VirtualBox* pour simuler un environnement de déploiement Linux. Du côté de la robotique, c'est la marque *Fanuc*, nous avons gardé les robots existants.



Figure 35 : Exemple de fournisseurs matériels

## 21.4 Choix des logiciels

- ❖ [MySQL](#) : Utilisé pour stocker et gérer les données sous forme de tableau.
- ❖ [Adminer](#) : Interface de visualisation de base de données SQL, compatible MySQL.
- ❖ [Figma](#) : Conception d'interface utilisateur (UI) et d'expérience utilisateur (UX) de prototypage, création de maquettes et de design interactif.
- ❖ [Visual Studio Code](#) : Un éditeur de code source, adapté à une variété de langages de programmation.
- ❖ [Looping](#) : Looping est un logiciel de modélisation conceptuelle de données.
- ❖ [Gantt Project](#) : Gestion de projet, suivi des tâches, gérer des projets de manière visuelle.
- ❖ [Github](#) : Plateforme de développement collaboratif de logiciels reposant sur le système de contrôle de version Git, permet l'hébergement de dépôts de code source.

## 21.5 Principaux langages

- ❖ [HTML 5](#) : Langage de balisage standard pour les pages web
- ❖ [CSS 3](#) : Langage permettant de mettre en forme des documents HTML
- ❖ [JavaScript](#) : Langage de programmation interprété
- ❖ [SQL](#) : Langage standardisé utilisé pour gérer des bases de données relationnelles
- ❖ [Pyqt5](#) : Bibliothèque Python qui permet de créer des interfaces graphiques (GUI) multiplateformes



Figure 36 : Exemple de langage de programmation

## 22 DÉVELOPPEMENT DU PROJET WEB

### 22.1 Application Web (Flask)

J'ai réalisé une interface Web. Elle comporte plusieurs pages et menu d'une dizaine de pages mais facilement ajustable. ([Voir Annexes](#)). Cette application doit être simple de compréhension pour tout opérateur ainsi que simple d'utilisation car elle sera destinée au pilotage de la machine et à la l'aiguillage des programmes de trajectoires robotiques.

#### Qu'est-ce que c'est ? :

Une Application légère est une interface Web utilisateur qui permet de connecter un opérateur et / ou un technicien à un système, une installation. L'utilisateur doit être en mesure d'interagir avec les différentes fonctionnalités mises en œuvre. Le processus industriel, et donc l'interface, doit être simple de lecture et d'utilisation pour les utilisateurs.

Nous organisons de ce fait des rencontres avec le client et les futurs utilisateurs pour définir les besoins. Cette interface web prendra en charge la gestion de l'installation, montage et vissage des portes avant, nous pouvons lire également les informations en instantané avec un court taux de rafraîchissement (variables). Les pages et menus sont organisés de façon suivante : 4 niveaux hiérarchiques d'accès : **Administrateurs, Conducteur d'Installation, Opérateurs, Commercial/Invité**, permet une meilleure gestion des autorisations. Les opérateurs ont la possibilité d'interagir et de visualiser les vis à contrôler, reprendre, le couple de vissages etc...

The screenshot shows a web-based alarm monitoring system. At the top, there's a navigation bar with links: Accueil, Alarmes, Statistiques, Paramètres, Appels CI, and a search bar. On the left, a sidebar lists several pages: Page Voiture, Page MEF, Page PAV, Page PAR, and Page AGV. The main content area is titled "Alarme i". It features three line graphs: "Declenchement by day of the week" (values 1, 1, 4, 9), "Declenchement by month of the year" (values 13, 1, 1), and "Declenchement by hour of the day" (values 4, 1, 6, 4). Below the graphs is a link to "Paramètres d'alarme". To the right of the graphs is a table of alarm logs:

alarme_id	id_code	nom	declenchement
1	1000	alarme_test	2023-11-21 10:30:00
2	143	Fault Robot 43	2023-01-20 08:00:00
3	144	Fault Robot 44	2023-02-15 10:30:00
4	142	Fault Robot 42	2024-01-11 06:55:50
5	141	Fault Robot 41	2024-01-11 06:56:23
14	1	Erreur Communication	2024-01-11 06:53:48
15	2000	Arrêt d'urgence station 1	2024-01-11 06:57:29

At the bottom of the page, a footer note reads: "2024 Stellantis-ARSN. Tous droits réservés."

*Figure 37 : Page visualisation Alarme applicatif web*

[Voir Annexes Pages Applicatif Web pour plus de visuels interfaces](#)

## 22.2 Programmation Application Web

### 22.2.1 Structure Applicatif Léger

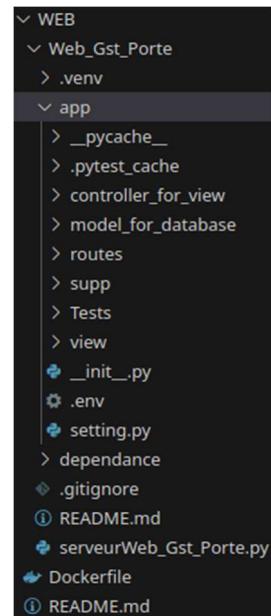
L'application légère est structurée selon le modèle **MVC** (*Modèle-Vue-Contrôleur*).

**model\_for\_database (Model)** : Logique métier de l'application, les classes traitent les données et les opérations sur les données.

**view (View)** : Interface utilisateur web **HTML & CSS**, permet les interactions de l'utilisateur.

**controller for view (Controller)** : Intermédiaire modèle-vue, traite les actions de l'utilisateur et met à jour le modèle en conséquence, rafraîchi la vue.

Le modèle MVC est une approche robuste pour organiser et développer de petites applications légères de manière modulaire, ce qui facilite la maintenance et l'évolutivité du code.



### 22.2.2 Environnements virtuels

L'environnement virtuel (**.venv**) crée un espace isolé où les dépendances spécifiques à un projet peuvent être installées sans affecter le système global. Cela assure la cohérence de l'environnement de développement et facilite la gestion des dépendances. Il est donc aisément de travailler d'un poste à un autre en évitant les conflits de dépendance. C'est pourquoi lors de déploiement, nous installerons uniquement les dépendances de l'environnement virtuel stockées dans le fichier « *requierments.txt* ».

### 22.2.3 Variables d'environnement

Dans notre projet, on utilise un fichier de configuration appelé (**.env**) pour stocker les variables d'environnement sensibles, telles que des clés secrètes ou des informations de connexion à la base de données. Ce qui permet une gestion standardisée et sécurisée de ces variables, évitant de les exposer directement dans le code source, cela renforce la sécurité et la portabilité de l'applicatif web. Cette méthodologie a été établie également pour l'applicatif lourd.

```
DB_HOST=172.  
DB_USER =  
DB_PASSWORD =  
DB_DATABASE=  
  
FLASK_SECRET_KEY=clé_secrète  
  
METABASE_SITE_URL = "http://:3000"  
METABASE_SECRET_KEY = "9b1305202dc5e"
```

Figure 38 : Exemple du fichier de variables d'environnement « .env »

### 22.2.4 Requêtes préparées ORM

Dans notre solution web, l'utilisation d'un **ORM** (*Object-Relational Mapping*) tel que **Peewee** est d'une importance capitale. Cette approche, nous permet de manipuler les données en nous affranchissant des complexités des requêtes SQL brutes. L'ORM simplifie le développement, garanti la cohérence et la sécurité des données (Injection SQL), assurer la portabilité de notre code d'applicatif.

## 22.3 Recommandation sécurité OWASP – ANSSI

L'OWASP-Top 10 propose des mesures proactives pour empêcher l'exploitation de ces failles de sécurité

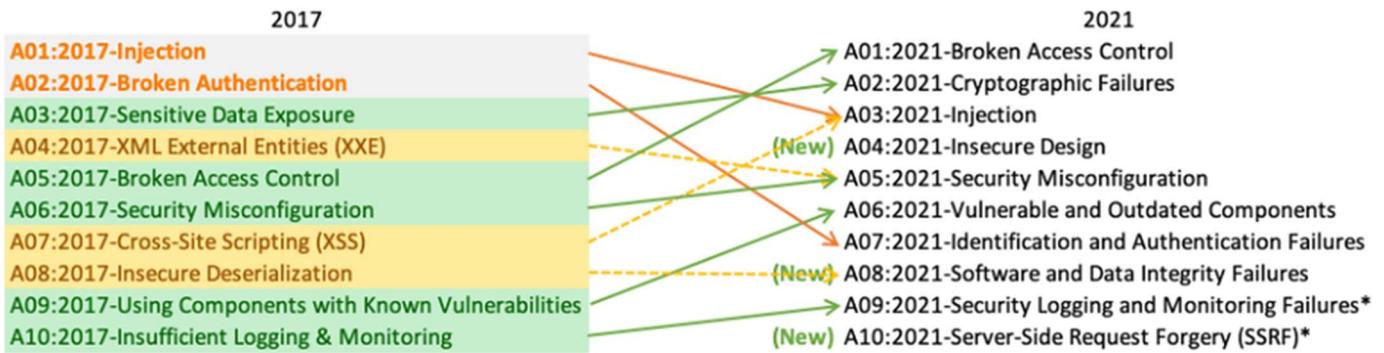


Figure 39 : Illustration capturé du site communautaire de l'OWASP : Recommandation TOP 10

### Par exemple :

Authentification : Implémentation d'un hachage de mot de passe fort avec des algorithmes sécurisés comme l'utilisation du Framework « *Flask\_Bcrypt* ».

```
utilisateur = Utilisateur.select().where(Utilisateur.identifiant == username).first()

if utilisateur:
    # Récupérer le mot de passe haché stocké dans la base de données
    mdp_bdd = utilisateur.mdp

    # Récupérer les 29 premiers caractères du mot de passe stocké dans la base de données (sel)
    sel = utilisateur.mdp[:29]

    # Hacher le mot de passe saisi par l'utilisateur avec le sel
    mdp_hache_utilisateur = generate_password_hash(password + sel)
```

Figure 40 : Extrait du hachage de mot de passe utilisateur Application Légère

Contrôle d'accès utilisateur : Implémentation des contrôles d'autorisation pour restreindre l'accès aux données et aux fonctionnalités en fonction du rôle et des permissions de l'utilisateur. Modifier les accès aux routes.

```
# ----- Configurations -----#
#----- Vérifier l'authentification de l'utilisateur -----
#-----#
bcrypt = Bcrypt()

def login_required(f):
    @wraps(f)
    def decorated_function(*args, **kwargs):
        print("Décorateur login_required : logged_in =", session.get('logged_in'))
        if not session.get('logged_in'):
            return redirect(url_for('auth'))
        return f(*args, **kwargs)
    return decorated_function
```

Figure 41 : Extrait fonction décoration « Brocken Access » des /route

```
@app.route('/home')
@login_required
def home():...
```

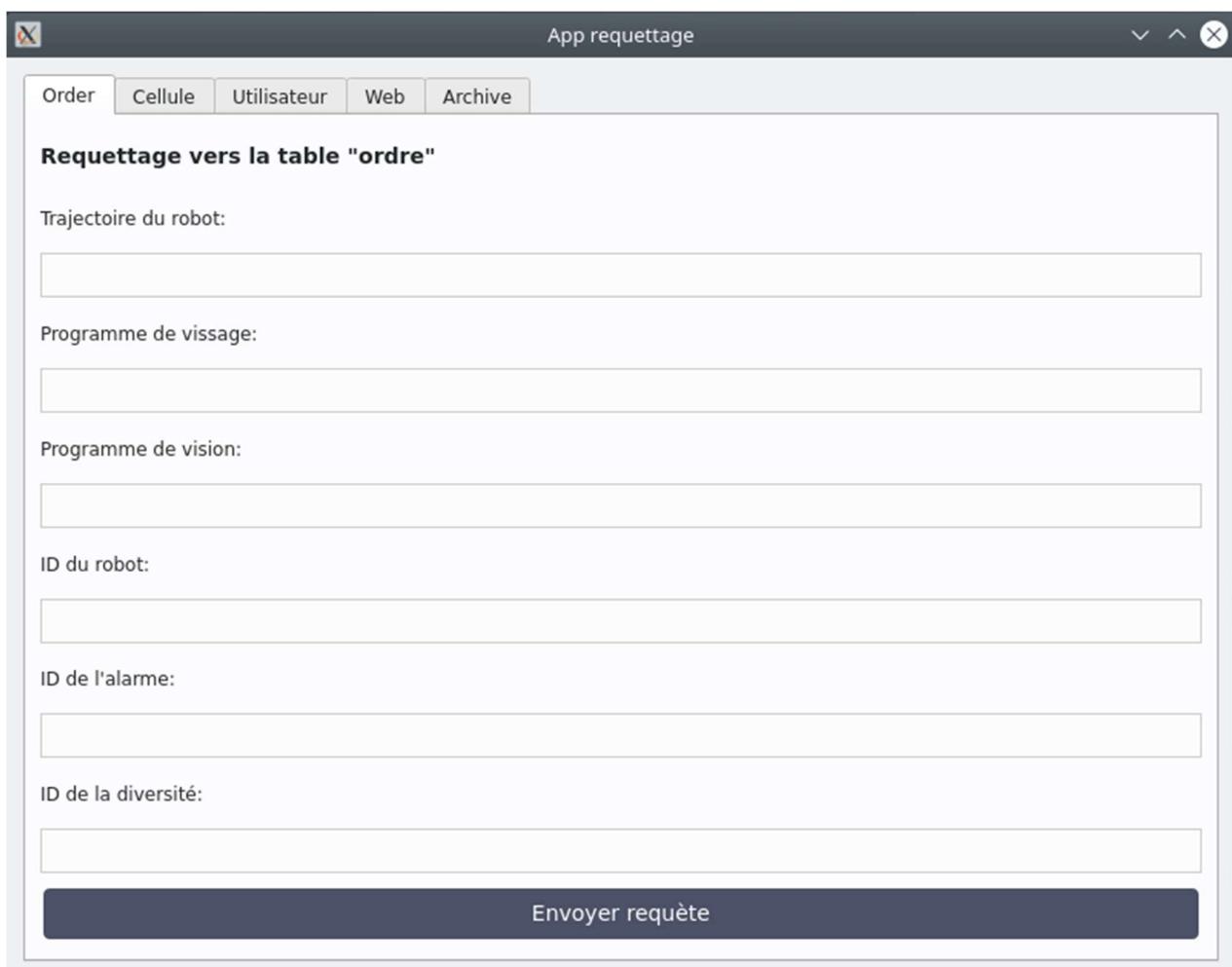
Figure 42 : Extrait décorateur sur la route « /home »

## 23 DÉVELOPPEMENT DU PROJET LOURD

### 23.1 Application Lourde (Pyqt5)

L'application lourde est développée avec **PyQt5** sur un système d'exploitation Linux, elle offre une interface simple permettant d'interagir avec une base de données. Elle permet aux utilisateurs de saisir des données dans des champs qui, une fois entrées, déclenchent des commandes d'aiguillage pour les robots ou alimente simplement la base avec d'autres données hors process. De plus, elle offre la possibilité de modifier les mots de passe pour une gestion sécurisée des accès.

Une fonctionnalité clé l'application est la visualisation de l'interface web, offrant une vue intuitive des informations de production en cours. De plus, elle intègre une option de sauvegarde de la base de données, accessible d'un simple clic, assurant la sécurité et la disponibilité des données critiques. En résumé, l'application lourde combine des fonctionnalités de gestion efficace et sécurisée des opérations liés au processus de l'installation, tout en offrant une expérience utilisateur simple et robuste.



The screenshot shows a window titled "App requettage". At the top, there is a menu bar with tabs: Order, Cellule, Utilisateur, Web, and Archive. The "Order" tab is currently selected. Below the menu, there is a section titled "Requettage vers la table \"ordre\"". This section contains several input fields for querying the database:

- Trajectoire du robot: An empty text input field.
- Programme de vissage: An empty text input field.
- Programme de vision: An empty text input field.
- ID du robot: An empty text input field.
- ID de l'alarme: An empty text input field.
- ID de la diversité: An empty text input field.

At the bottom of the window is a dark blue button labeled "Envoyer requête" (Send query).

Figure 43 : Page Requête applicatif lourd

Voir Annexes Pages Applicatif lourd pour plus de visuels interfaces

## 23.2 Programmation Application Lourde

### 23.2.1 Structure Applicatif Lourd

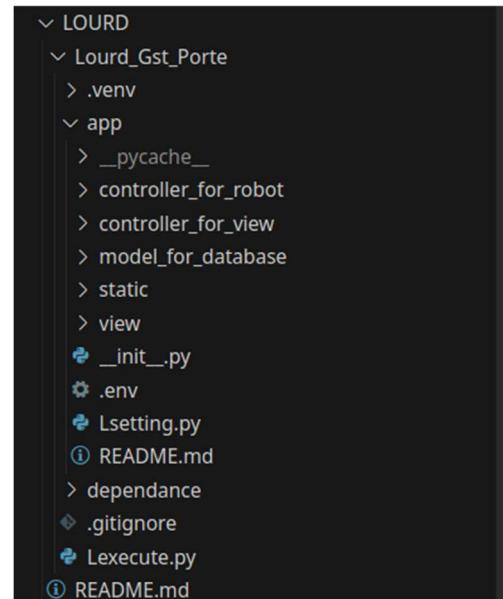
L'application est développée avec **PyQt5** et suit le modèle **MVC**, cela signifie une structure de code permettant de séparer les préoccupations liées aux données (le modèle), à l'interface utilisateur (la vue), et à la logique de contrôle (le contrôleur). Nous avons repris le MVC souvent utilisé en web pour rester organisé, en séparant les différentes couches on favorise la réutilisation du code

**model for database (Model)** : structure des données manipulées

**view (View)**: interfaces utilisateur, widgets PyQt5

**controller for view (Controller)** : interactions utilisateur, traite les événements et met à jour le modèle

**controller for robot (Controller)** : interactions robot, traite ordre et mouvement robotique



### 23.2.2 Fonction de mouvement Robot

```
from fanucpy import Robot

def start_robot_move(trajectory):
    robot = Robot(
        robot_model="Fanuc",
        host="172.16.1.100",
        port=18735,
        ee_D0_type="RDO",
        ee_D0_num=7,
    )
    robot.connect()

    print(f"Debut TOP : {robot.get_rdo(1)}")
    print(f"Current pose: {robot.get_curpos()}")
    print(f"Current joints: {robot.get_curjpos()}")
    print(f"Instantaneous power: {robot.get_ins_power()}")

    robot.set_rdo(rdo_num=1, val=True)
    print(f"TOP: {robot.get_rdo(1)}")

    print("Nom trajectoire: " ,trajectory)
    robot.call_prog(trajectory)
    print("Run program: " ,trajectory)

    print(f"Fin TOP: {robot.get_rdo(1)}")
    print(f"Current pose: {robot.get_curpos()}")
    print(f"Current joints: {robot.get_curjpos()}")
    print(f"Instantaneous power: {robot.get_ins_power()}")

    robot.disconnect()
```

Le code ci-joint est la séquence et les méthodes pour envoyer un ordre de trajectoire au robot industriel. Il est composé de 5 grandes étapes.

- Etp1. Import de la classe « Robot »
- Etp2. Définition des configuration robot
- Etp3. Connexion
- Etp4. Ordre/Retour programme
- Etp5. Déconnexion

*L'adresse du robot, ports et autres configurations robots sont en clair pour la partie pré-déploiement, ils seront stockés dans un fichier (.env) pour le déploiement de production.*

*Figure 47 : Extrait fonction « robot\_move.py »*

### 23.2.3 Fonction Widget / Interface Utilisateur

```
class OrderApp(QWidget):
    def __init__(self):
        super().__init__()
        self.init_ui()

        # Signal méthode de démarrage du mouvement du robot
        self.order_tab.trajectoire_envoyee.connect(self.start_robot_move)

    def start_robot_move(self, trajectoire):
        #Fonction start_robot_move de robot_move.py avec la trajectoire spécifiée
        robot_move.start_robot_move(trajectoire)

    def init_ui(self):
        self.setWindowTitle('App requettage')
        self.setGeometry(300, 250, 800, 600) # Ajuster la largeur pour les onglets

        # Création d'un QTabWidget pour gérer les onglets
        tab_widget = QTabWidget(self)

        # Ajout d'onglets
        self.order_tab = OrderTab()
        cellule_tab = CelluleTab()
        utilisateur_tab = UtilisateurTab()
        web_tab = WebTab()
        archive_tab = ArchiveTab()

        tab_widget.addTab(self.order_tab, 'Order')
        tab_widget.addTab(cellule_tab, 'Cellule')
        tab_widget.addTab(utilisateur_tab, 'Utilisateur')
        tab_widget.addTab(web_tab, 'Web')
        tab_widget.addTab(archive_tab, 'Archive')

        # Mise en page principale
        main_layout = QVBoxLayout(self)
        main_layout.addWidget(tab_widget)
```

Figure 48 : Extrait programme « order\_app.py »

Ce code crée une application graphique en utilisant PyQt5, une bibliothèque Python pour créer des interfaces d'un ou des utilisateurs. L'application comporte plusieurs onglets gérés par un *QTabWidget*. Chaque onglet est une partie différente de l'application.

La classe principale est *OrderApp*, qui hérite de *QWidget*. Dans son constructeur *init()*, elle initialise l'interface utilisateur en appelant la méthode « *init\_ui()* ». Elle connecte également un signal provenant de l'onglet de commande (*self.order\_tab*) à une fonction « *start\_robot\_move()* ». Lorsque ce signal est émis, la fonction « *start\_robot\_move()* » est appelée, envoyant une trajectoire spécifiée à une fonction externe (*start\_robot\_move*) située dans un fichier appelé *robot\_move.py*. Cette fonction externe démarre le mouvement d'un robot avec la trajectoire spécifiée.

En résumé, l'application crée une interface utilisateur avec des onglets pour différentes fonctionnalités et permet de démarrer le mouvement d'un robot en fonction des commandes entrées dans l'onglet correspondant.

## 23.3 Simulation et implantation mécanique

J'ai réalisé un programme structuré pour la réalisation des mouvements, de prises robot et de vissage dans le but, de réaliser une simulation de production et d'établir un temps de cycle ainsi que de vérifier le bon fonctionnement des applicatifs en interaction avec le métier robotique (robots, ensemble et sous-ensembles).

Nous utiliserons **RoboGuide** qui est le logiciel de Fanuc pour répondre au cahier des charges client. En parallèle, nous utiliserons un package python (**pyfanuc**) qui nous simplifiera la communication entre l'applicatif et les baies robots.

FANUC ROBOGUIDE est un simulateur de robots qui simule les commandes de mouvement et d'application du robot. Pour garantir un minimum d'impact sur la production et pour gagner du temps dans la modélisation 3-D, les modèles de pièces peuvent être importés à partir d'un PC sous forme de données CAO. C'est pourquoi, les projeteurs mécaniques réalisent la structure et l'implantation de la machine ce qui nous permet de les tester et de les modifier. Le contrôleur virtuel interne permet de simuler les trajectoires, de calculer et valider les temps de cycle et d'éviter toutes collisions par anticipation. RoboGuide est intuitif et extrêmement facile d'utilisation.

- ➔ Rappel des contraintes principales : Revissage de porte
- ➔ Rappel des contraintes secondaires : Interface web dans la console (Teach Robot)

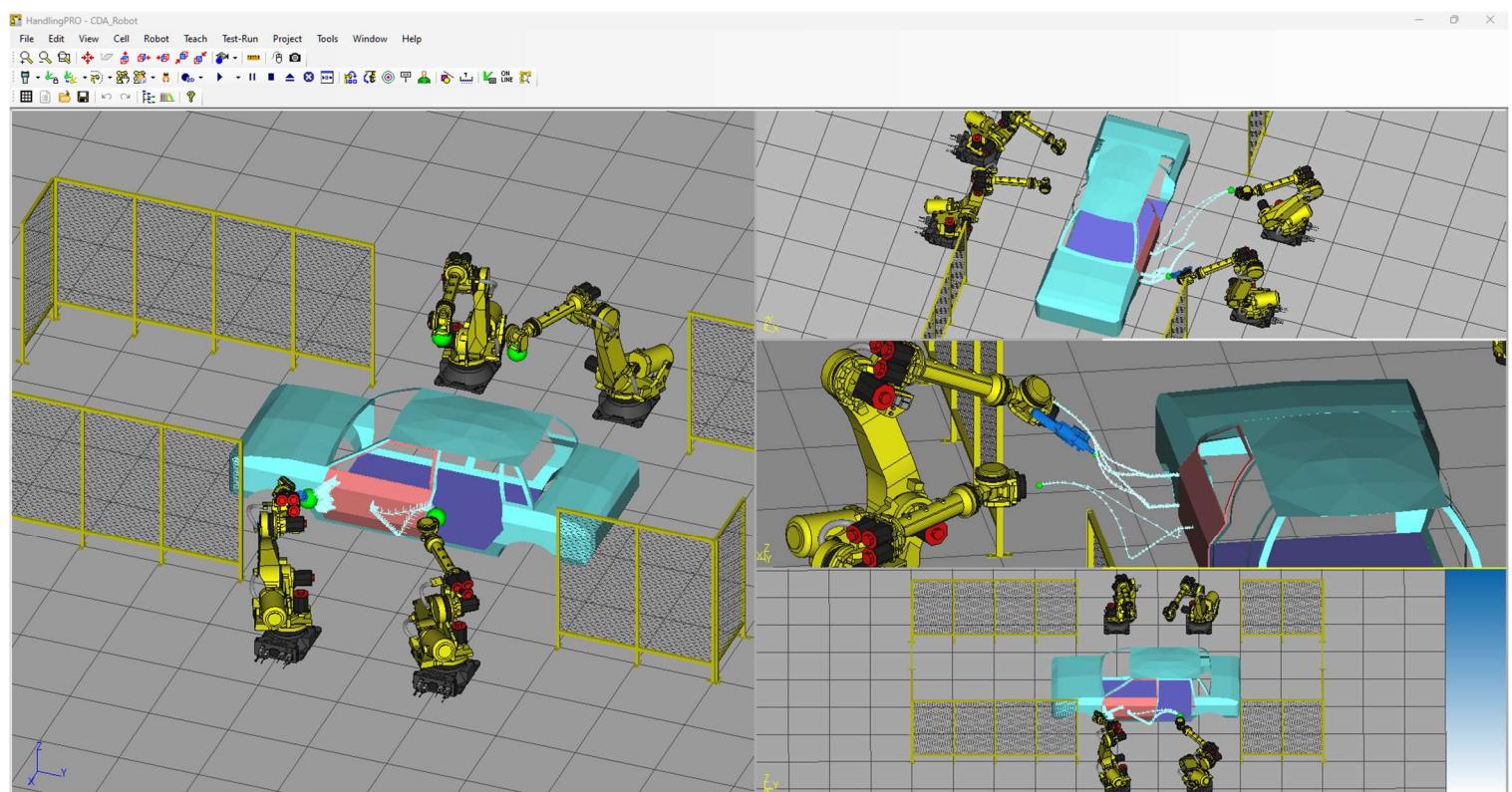


Figure 44 : Interface de simulation RoboGuide

**Voir Annexes App Web visuels sur l'interface Teach Pendant**

## 23.4 Table d'échanges

Avant d'effectuer les programmes automates et robots, nous créons un fichier Excel où nous listons toutes les IO, registres, positions registre, les constantes variables et applications. Ces informations seront très importantes pour la suite.

## 23.5 Communication entre Applicatif Lourd - Robot

Dans le développement de ma communication, je vais utiliser un package python déjà existant ce qui simplifiera le déploiement. Le package est composé de deux parties :

1. Code d'interface du robot écrit en langage de programmation Python.
2. Pilote de contrôleur de robot FANUC (testé avec le contrôleur R-30iB Mate Plus) écrit en KAREL et FANUC enseigner les langages du pendentif

Le protocole de communication entre le package Python et le contrôleur du robot FANUC est illustré ci-dessous :

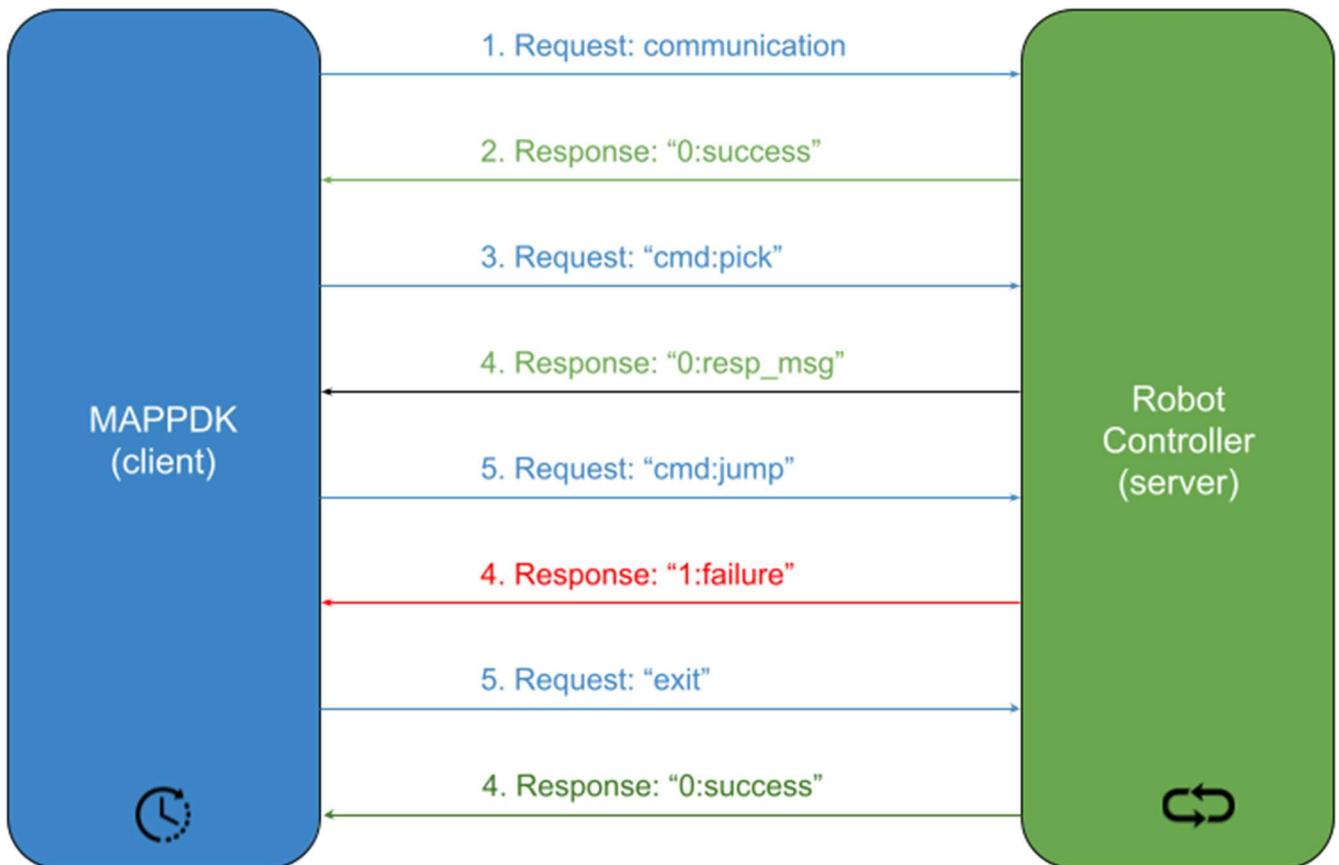


Figure 45 : Schématique de communication Robot serveur et AppLourde

## 23.6 Programmation du Robot conformatiion /vissage

Les programmes **TPE** sont écrits pour gérer les points de trajectoire et piloter les actionneurs, c'est un langage machine propre à la marque Fanuc. J'ai réalisé un programme **KAREL** structuré pour la réalisation de gestion du robot, c'est ici que nous échangeons les informations entre l'applicatif et le robot. Pour cela, nous utilisons le logiciel Visual Code.

### 23.6.1 Programme principal d'aiguillage

Nous utiliserons le logiciel RoboGuide pour écrire le programme principal, dit « T\_REVIS » pour l'aiguillage des robots de vissage vers les positions demandées. Le principe étant d'appeler les actions à réaliser par le robot, les points, les trajectoires et quelques contrôles de capteurs à cet endroit que l'on dédiera plutôt aux étapes. Dans cette étape, nous programmons purement avec des fonctions logiques simples (IF THEN ; WHILE ; CALL) et quelques équations.

Le Top départ de ce programme « T\_REVIS » est défini par le programme « mappdk\_server.pc » (fichier serveur) & « mappdk\_logger.pc » (fichier de connexion). Sous la forme d'un langage Karel. Le fichier principal MAPPDK « mappdk.ls » qui doit être exécuté à partir du Teach, fait tourner en tache de fond ces deux fichiers.

```

R41-VISSAGE ▾ T_REVIS ▾
Busy Step Hold Fault
Run I/O Prod TCyc T_REVIS LINE 0 AUTO ABORTED JOINT 100 %
ROBOT41 T_REVIS 1/21
1: !----CONFIGURATION----
2: UFRAME_NUM=8
3: UTTOOL_NUM=8
4:
5: !----TOP APPLICATION----
6: IF RO[1]=ON,JMP LBL[1]
7: LBL[1]
8:
9: !----PRENDRE ZONE ITV----
10: WAIT (DI[100])=ON
11:
12: !Trajectoire de vissage 1-4
13: CALL T_L1
14: CALL T_L2
15: CALL T_L3
16: CALL T_L4
17: !----ZONE RENDU ITV----
18: DO[101]=ON
19: WAIT 1.00(sec)
20: DO[101]=OFF

```

Figure 46 : Interface Teach pendant du programme principal « A\_ROBOT1 »

### 23.6.2 Programme de mouvement

Comme indiqué précédemment, les trajectoires sont programmées sous formes de sous-programme. Pour cela, nous utiliserons des fonctions logiques simples. Les points sont appris sur les axes de coordonnées cartésiennes.

## 24 PLAN DE TESTS DES APPLICATIFS

Les Tests Unitaires/Intégration/Fonctionnels sont réalisés lors du pré-déploiement. Les Tests Sécurité/Charges/Performances sont réalisés lors du post-déploiement. Nous avons fait le choix d'isoler nos tests et de ne pas les fournir aux clients. *C'est donc dans ce contexte que nous nous permettons d'inscrire en clair les adresses IP et identifiants, mdp de préproduction. Les informations sensibles sont donc invalides à la mise en production.*

### 24.1 Plan de tests application lourde

Type de Test	Objectif	Outils	Cas/Scénarios
<b>Unitaires</b>	Vérification des Composants Individuels	Pytest	Fonctions individuelles Assure les mouvements robot
<b>Intégration</b>	Tester le processus	Manuel/Roboguide	Vérification informations robots live
<b>Fonctionnels</b>	Vérifier la Fonctionnalité	Manuel/Roboguide	Aiguillage robot en fonction des champs saisis par l'utilisateur
<b>Sécurité</b>	Identifier les Vulnérabilités	QIntValidator/QMessageBox	Validation entrées utilisateur & fuites informations
<b>Charges</b>	Évaluer les Performances & Capacité de charge	Manuel	Plusieurs envoient ordres simultané
<b>Performances</b>	Évaluer les Performances	Manuel	Analyse des performances

### 24.2 Plan de tests application légère

Type de Test	Objectif	Outils	Cas/Scénarios
<b>Unitaires</b>	Vérification des Composants Individuels	Pytest	Fonctions individuelles Assure la connexion à base de données
<b>Intégration</b>	Tester le processus	Pytest	Vérification processus d'authentification à l'application
<b>Fonctionnels</b>	Vérifier la Fonctionnalité	Pytest / Selenium	Automatisation connexion opérateur application Injection SQL, XSS, CSRF, Gestion des sessions, contrôle des accès & fuite données sensibles
<b>Sécurité</b>	Identifier les Vulnérabilités	ZAP OWASP	Gestion des sessions, contrôle des accès & fuite données sensibles
<b>Charges</b>	Évaluer les Performances & Capacité de charge	App Lourde	Requête envoyée simultanément
<b>Performances</b>	Évaluer les Performances	Lighthouse Chromium	Analyse des performances

## 25 TESTS DE L'APPLICATIF WEB DE PRE-DEPLOIEMENT

### 24.3 Tests Unitaires

Dans le cadre du développement de l'application Web, fonctionnant sur une base de données, il est essentiel de garantir le bon fonctionnement de la fonctionnalité de connexion. Les tests unitaires ont été réalisés dans un environnement de développement, pré-déploiement. Nous avons utilisé **Pytest**, un Framework de test en Python, pour écrire et exécuter les tests.

L'objectif de ce test est d'assurer la qualité et la fiabilité de connexion de mon application à la base de données MySQL ainsi que de retrouver un champ spécifique dans une table.

**test\_connect\_success** : Ce test vérifie que la méthode « connect » réussit à établir une connexion à la base de données lorsque des identifiants valides sont fournis

**test\_connect\_failure** : Ce test vérifie que la méthode « connect » échoue à établir une connexion à la base de données lorsque des identifiants invalides sont fournis

**test\_gettable** : Ce test vérifie que la méthode « gettable » récupère correctement les données de la table "utilisateur" après une connexion réussie à la base de données.

```
import pytest
from if_bdd import IF_bdd

@pytest.fixture
def bdd():
    return IF_bdd()

def test_connect_success(bdd):
    assert bdd.connect('172. [REDACTED]', 'user', 'password', 'Gst_[REDACTED]') is True

def test_connect_failure(bdd):
    assert bdd.connect('172. [REDACTED]', 'invalid_user', 'invalid_password', 'Gst_[REDACTED]') is False

def test_gettable(bdd):
    # Connexion à la base de données avec l'utilisateur factice
    bdd.connect('172. [REDACTED]', 'test_user', 'test_password', 'Gst_[REDACTED]')

    # Appel à la méthode gettable avec la table "utilisateur"
    result = bdd.gettable('utilisateur')

    # Vérification des résultats
    assert result == [(17, ' [REDACTED]', '$2b$12$ [REDACTED'] F1Fdf9ccSo9V44GWFIUU6')] [REDACTED]
```

Figure 49 : Test unitaire connexion base de données SQL « test\_if\_bdd.py »

Le message « collected 3 items » indique que **pytest** a collecté avec succès 3 éléments de test à exécuter. Chacune de ces fonctions est un cas de test individuel qui sera exécuté par pytest

```
user@uimm:~/Documents/CDA/WEB/Web_Gst_Porte/app/Test_Unitaire$ pytest test_if_bdd.py
=====
platform linux -- Python 3.9.2, pytest-8.1.1, pluggy-1.4.0
rootdir: /home/user/Documents/CDA/WEB/Web_Gst_Porte/app/Test_Unitaire
collected 3 items

test_if_bdd.py [REDACTED]
```

Figure 50 : Résultat du test unitaire « test\_if\_bdd.py »

## 24.4 Test d'Intégration

Ce test d'intégration vérifie le comportement de l'application lorsque l'authentification réussie.

**test\_auth\_success** : Ce test vérifie le comportement de l'application lorsque l'authentification réussit. Il simule une requête POST vers l'URL « /auth » avec des identifiants valides. Ensuite, il vérifie que la réponse de l'application à un code de statut HTTP 302, ce qui indique une redirection.

**response.status\_code == 302** : Cette assertion vérifie que la réponse de l'application est une redirection. Le code de statut HTTP 302 est utilisé pour indiquer une redirection vers une autre URL après une action réussie, comme l'authentification réussie dans ce cas.

Si l'authentification réussie, l'utilisateur est redirigé vers une autre page de l'application. La redirection est un comportement attendu après une authentification réussie, car l'application redirige l'utilisateur vers une page d'accueil après connexion.

```
import pytest

from app import app

@pytest.fixture
def client():
    with app.test_client() as client:
        yield client

def test_auth_success(client):
    # Simuler une requête POST pour l'authentification réussie
    response = client.post('/auth', data={'username': '██████████', 'password': '████████'})
    assert response.status_code == 302 # Vérifie que la redirection est effectuée après l'authentification réussie
```

Figure 51 : Test intégration authentification application « test\_integration\_auth.py »

Le test a réussi, ce qui signifie que le processus d'authentification fonctionne correctement lorsque des identifiants valides sont fournis. Cela est confirmé par le code de statut HTTP 302 renvoyé par l'application.

```
user@uimm:~/Documents/CDA/WEB/Web_Gst_Porte/app$ pytest test_integration_auth.py
=====
platform linux -- Python 3.9.2, pytest-8.1.1, pluggy-1.4.0
rootdir: /home/user/Documents/CDA/WEB/Web_Gst_Porte/app
collected 1 item

test_integration_auth.py .

=====
1 passed in 0.83s
```

Figure 52 : Résultat test intégration « test\_integration\_auth.py »

## 24.5 Test Fonctionnel

Ce code est un exemple d'un test d'automatisation d'authentification utilisant **Selenium** et **pytest**.

Selenium est un ensemble d'outils et de bibliothèques open-source. Permet créer des scripts qui interagissent avec les navigateurs web de la même manière qu'un utilisateur humain le ferait

**test\_authentication** : Utilise la fixture *browser* pour obtenir l'objet du navigateur, ici on utilise **Chromium**. Navigue vers la page de connexion « /auth ». Remplit le champ d'identifiant avec le texte "Mon2Identifiant". Remplit le champ de mot de passe avec le texte "Mon2mdp". Clique sur le bouton de soumission. Vérifie que l'URL actuelle du navigateur est celle de la page d'accueil après l'authentification « /home ».

```
import pytest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

@pytest.fixture
def browser():
    # Initialiser le navigateur Selenium avec ChromeDriver
    driver = webdriver.Chrome()
    yield driver
    # Fermer le navigateur une fois le test terminé
    driver.quit()

def test_authentication(browser):
    # Ouvrir la page de connexion
    browser.get("http://[REDACTED]/auth")

    # Attendre que l'élément avec l'ID "username" soit visible
    username_input = WebDriverWait(browser, 10).until(
        EC.visibility_of_element_located((By.ID, "username"))
    )
    username_input.send_keys("Mon2Identifiant")

    # Attendre
    password_input = WebDriverWait(browser, 10).until(
        EC.visibility_of_element_located((By.ID, "password"))
    )
    password_input.send_keys("Mon2mdp")

    # Attendre
    submit_button = WebDriverWait(browser, 10).until(
        EC.element_to_be_clickable((By.CSS_SELECTOR, "button[type='submit']"))
    )
    submit_button.click()

    # Vérifier que l'utilisateur est redirigé vers la page d'accueil
    assert browser.current_url == "http://[REDACTED]/home"
```

Figure 53 : Test fonctionnel connexion application web « *test\_fonctionnel\_auth.py* »

Le test a réussi, ce qui signifie que le processus d'authentification fonctionne correctement lorsque des identifiants valides sont fournis. Cela est confirmé par l'automatisation des étapes de connexion par l'API.

```
user@uimm:~/Documents/CDA/WEB/Web_Gst_Porte/app$ pytest test_fonctionnel_auth.py
=====
platform linux -- Python 3.9.2, pytest-8.1.1, pluggy-1.4.0
rootdir: /home/user/Documents/CDA/WEB/Web_Gst_Porte/app
collected 1 item

test_fonctionnel_auth.py .
[100%]
=====
1 passed in 4.67s
```

Figure 54 : Résultat test fonctionnel « *test\_fonctionnel\_auth.py* »

## 26 DEPLOYEMENT DES APPLICATIFS

### 26.1 Plan de déploiement Applicatif Web

#### 26.1.1 Préparation

<b>1</b>	Accessibilité au Project Github et KeePass
<b>2</b>	Readme de déploiement
<b>3</b>	Configuration de l'environnement

On s'assure d'avoir les prérequis par exemple l'os et la version, via le terminal :

```
user@uimm:~$ uname -o
GNU/Linux
user@uimm:~$ uname -v
#1 SMP Debian 5.10.209-2 (2024-01-31)
```

Docker et Docker Compose doivent être installés sur le système on peut vérifier via le terminal :

```
-----
Python 3.9.2
pip 20.3.4 from /usr/lib/python3/dist-packages/pip (python 3.9)
openjdk 11.0.22 2024-01-16
gcc (Debian 10.2.1-6) 10.2.1 20210110
g++ (Debian 10.2.1-6) 10.2.1 20210110
Docker version 26.0.0, build 2ae903e
Docker Compose version v2.25.0
=====
```

Il est également primordial de configurer son fichier (.env) pour le paramétrage des connexions inter-application et base de données ainsi que les clés secrètes.

#### 26.1.2 Déploiement

<b>1</b>	Déploiement des Stacks et montage automatique des images
<b>2</b>	Migration archive (.db) de la base de données SQL
<b>3</b>	Migration ou édition des tableaux de bord Metabase
<b>4</b>	Sauvegarde Portainer (snapshots des volumes et système fichiers container)

##### 26.1.2.1 Dockerisation applicatif web

Docker est une plateforme open-source qui automatisise le déploiement des applications dans des conteneurs logiciels. Ces conteneurs sont des environnements légers et portables qui contiennent tout ce dont une application a besoin pour s'exécuter, y compris le code, les bibliothèques systèmes, les outils systèmes. Les avantages de Docker résident dans sa portabilité, sa facilité de déploiement, son isolation des ressources et sa cohérence entre les environnements de développement, de test et de production. Nous utiliserons « [Portainer.io](#) » pour la gestion de nos conteneurs.

**Portainer** : Interface graphique open-source pour la gestion des conteneurs Docker. Il permet de gérer facilement les conteneurs Docker, les images, les volumes et les réseaux via une interface utilisateur web conviviale.

- Création de l'image :** Pour construire l'image de notre application web, nous devons nous assurer que notre application Flask est correctement structurée avec toutes les dépendances répertoriées dans le fichier requirements\_venv.txt. Ensuite, nous créons un fichier Dockerfile contenant les instructions nécessaires pour construire l'image Docker de notre application.

```
# -----DOCKFILE-----#
# Image Python comme image de base
FROM python:3.9-slim-bullseye

# Répertoire de travail dans le conteneur
WORKDIR /WEB

# Contenu actuel du répertoire dans le conteneur au répertoire de travail
COPY .

# Installation des dépendances nécessaires
RUN pip install -r Web_Gst_Porte/dependance/requirements_venv.txt

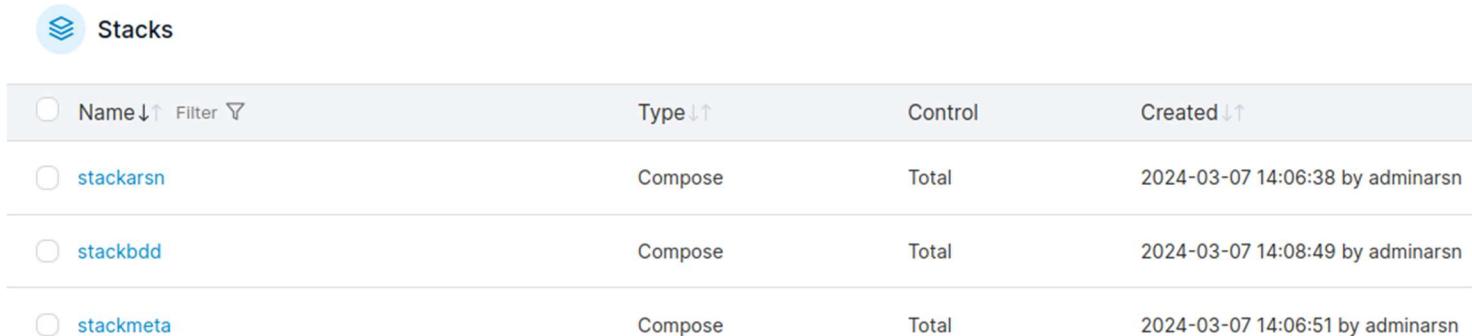
# Exposition port sur lequel l'application Flask écoute
EXPOSE 5000

CMD ["python3", "Web_Gst_Porte/serveurWeb_Gst_Porte.py"]

# -----INFOS-----#
LABEL version="0.1.9" maintainer="LOTODE Morgan <lotode.morganrsa@gmail.com>"
```

Figure 55 : Extrait Visual Code du fichier DockerFile

- Déploiement sur Portainer :** Une fois que notre image Docker a été testée avec succès localement, nous pouvons la déployer sur Portainer. Par l'interface utilisateur de Portainer dans le navigateur web, nous importons les stacks (**ensemble de services**) nécessaires à la configuration des réseaux et des services. Les stacks nous permettent d'automatiser le déploiement de nos conteneurs.



	Name ↓↑ Filter ▾	Type ↓↑	Control	Created ↓↑
<input type="checkbox"/>	stackarsn	Compose	Total	2024-03-07 14:06:38 by adminarsn
<input type="checkbox"/>	stackbdd	Compose	Total	2024-03-07 14:08:49 by adminarsn
<input type="checkbox"/>	stackmeta	Compose	Total	2024-03-07 14:06:51 by adminarsn

Figure 56 : Capture écran Portainer des stacks déployés

La configuration des stacks spécifie les services pour nos applications, et définit les paramètres requis, tels que les variables d'environnement, les volumes, les ports exposés, etc.

```

1  version: '2'
2
3  services:
4    flask_app_arsn:
5      image: flask_app_arsn:v0.1.9
6      container_name: flask_app_arsn
7      restart: always
8      ports:
9        - 5000:5000
10     networks:
11       mynetwork:
12         ipv4_address: 172.18.0.5
13
14   volumes:
15     mysql-volume:
16
17   networks:
18     mynetwork:
19       driver: bridge
20       ipam:
21         config:
22           - subnet: 172.18.0.0/16

```

Figure 57 : Édition du stack « Flask\_app\_arsn »

Ci-dessus, on peut voir le nom de l'image « flask\_app\_arsn » et la version « v0.1.9 », le port sur écoute « 5000 ». Nous avons décidé de fixer les adresses IP (avec une plage limite de 16 adresses) de nos applicatifs, pour une meilleure gestion et contrôle. Nous prêtons attention également à la configuration réseaux. Les 3 stacks communiquent entre eux et nécessitent donc une configuration réseau par pont.

3. **Vérification du déploiement :** Une fois le déploiement terminé, nous vérifions que notre application Flask est accessible via le port spécifié dans Portainer. Nous testons toutes les fonctionnalités de notre application pour nous assurer qu'elle fonctionne correctement dans l'environnement Docker.

Containers

<input type="checkbox"/>	Name ↴	State ↴ Filter ▾	Quick Actions	Stack ↴	Image ↴
<input type="checkbox"/>	adminer	running		stackbdd	adminer
<input type="checkbox"/>	flask_app_arsn	running		stackarsn	flask_app_arsn:v0.1.9
<input type="checkbox"/>	metabase	running		stackmeta	metabase/metabase
<input type="checkbox"/>	mysql	running		stackbdd	mysql
<input type="checkbox"/>	portainer	running		-	portainer/portainer-ce

Figure 58 : Capture écran Portainer de la liste des conteneurs déployés et en fonctionnement

4. **Maintenance et surveillance :** Une fois l'application déployée, nous surveillons son fonctionnement via les logs (journaux) de Portainer et effectuons toute maintenance nécessaire pour assurer un fonctionnement optimal.

**WARNING:** This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.

- \* Running on all addresses (0.0.0.0)
- \* Running on http://127.0.0.1:5000
- \* Running on http://172.18.0.5:5000

Press CTRI+C to quit

**Figure 59 : Journal (Logs) du conteneur « Flask\_app\_arsn »**

Ici, nous prenons l'exemple des journaux de l'applicatif Flask\_app\_arsn. Pour la phase de tests des Dockerisations des conteneurs, nous avons fait le choix de laisser par défaut l'écoute sur toutes les interfaces réseau disponibles sur la machine hôte.

### 26.1.3 Post-Déploiement

<b>1</b>	Ajout Utilisateurs et mot de passe
<b>2</b>	Tests de sécurité
<b>3</b>	Tests de charge
<b>4</b>	Tests de performance

Les tests de post-déploiement sont cruciaux pour évaluer la qualité et la performance de l'application une fois déployée. Ils permettent de détecter rapidement les problèmes éventuels introduits lors du déploiement, garantissant ainsi une expérience utilisateur optimale.

Ces tests valident également que toutes les fonctionnalités de l'application fonctionnent correctement dans *l'environnement de production*, assurant ainsi la cohérence et la fiabilité de l'expérience utilisateur. De plus, ils contribuent à identifier les goulots d'étranglement et les problèmes de performance, permettant ainsi d'optimiser la réactivité et la disponibilité de l'application.

## 27 TESTS DE L'APPLICATIF WEB POST-DEPLOIEMENT

### 26.1 Test de charge

Ce code est un test de charge réalisé avec le Framework **Locust** pour simuler le comportement des utilisateurs sur notre application web. Ici, on effectue des tâches comme l'accès à la page « /home » ou encore « /warning »

```
from locust import HttpUser, task, between

class AuthUser(HttpUser):
    wait_time = between(1, 5) # attente entre les requêtes

    @task
    def login_valid(self):
        # POST authentification valides
        response = self.client.post("/auth", data={"username": "████████", "password": "████████"})
        # Vérification
        if response.status_code == 200:
            print("Authentification réussie avec des identifiants valides")
        else:
            print("Échec de l'authentification avec des identifiants valides")

    @task
    def access_home_page(self):
        # accueil
        self.client.get("/home")

    @task
    def access_warning_page(self):
        # warning
        self.client.get("/warning")
```

Figure 60 : Test de charge application web « locustfile.py »

Les résultats ci-dessous de **Locust** fournissent une vue d'ensemble de la performance, en mettant en évidence les temps de réponse, le nombre de requêtes réussies et échouées, ainsi que les taux de requêtes et d'échecs par seconde. Ici, on peut remarquer qu'il n'a pas eu d'échecs.

Nous avons configuré le test pour 100 utilisateurs avec 1 à 5 secondes entre les requêtes sur une plage de 3 minutes.

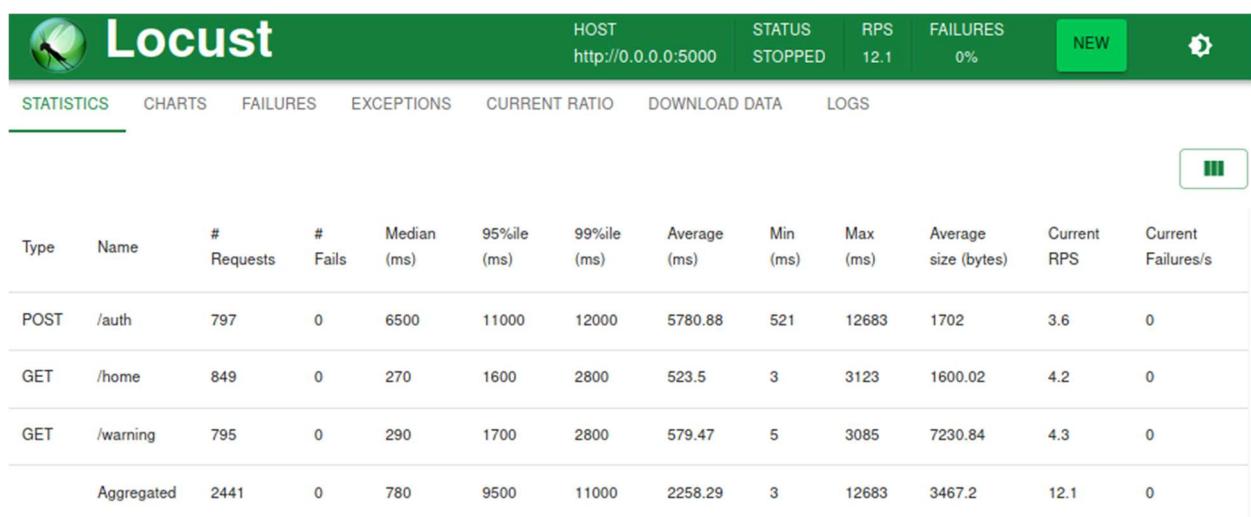


Figure 61 : Résultat interface Locust du test de charge « locustfile.py »

Le graphique de **Locust** présente une représentation visuelle de la charge de travail exercée sous test ainsi que des performances de l'application (extrait de l'application). En combinant ces données fournies par **Locust**, nous pouvons obtenir un extrait de la performance du système sous test et identifier les zones à optimiser ou à corriger.

**Figure 62 : Résultat graphique Locust du test de charge**

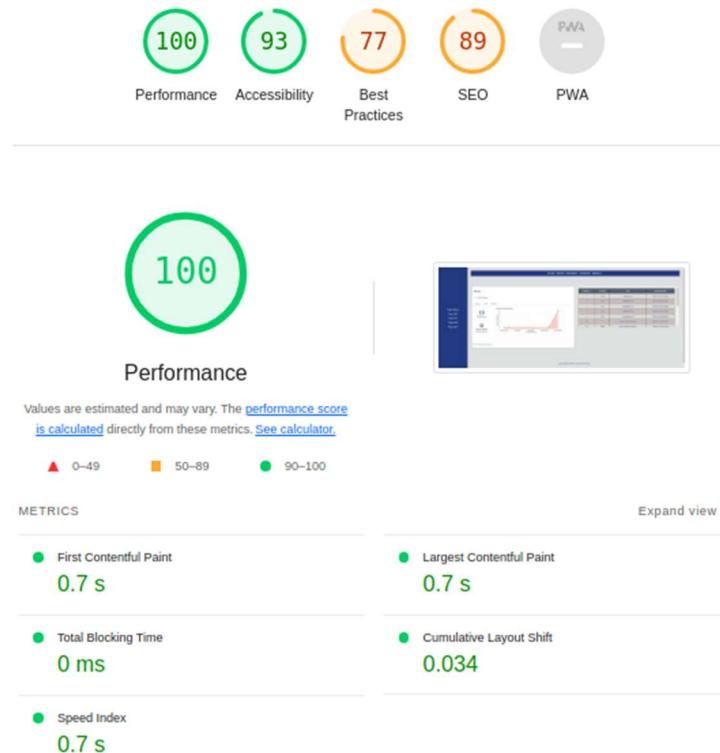


## 26.2 Tests de performance

On soumet notre site à un test sur Google Lighthouse par Chromium qui mesure la vitesse de chargement d'une page web et la rapidité avec laquelle les utilisateurs peuvent y accéder. C'est un moyen pour connaître les forces et faiblesses de son application web.

Les scores obtenus sont plutôt bons sur l'expérience utilisateur ainsi que sur son accessibilité adaptée à un grand nombre d'utilisateurs. On retient un score moyen sur l'analyse des bonnes pratiques, Google considère que nous ne sommes pas 100% conforme aux normes actuelles. Cela est compréhensible sachant que nous n'utilisons pas de HTTPS (application interne à l'entreprise, ne nécessite pas de déploiement sur le réseau Google).

En résumé, les scores démontrent que l'application web est performante, accessible pour l'utilisation finale.



**Figure 63 : Résultat interface Chromium Lighthouse de l'applicatif web**

## 26.3 Tests de sécurité & vulnérabilités

On trouve deux sites de référence sur la cybersécurité :

- **OWASP** (Open Web Application Security Project), Supporté par une communauté dédiée à la sécurité
- **ANSSI** (Agence Nationale de Sécurité des Systèmes d'Information), Edicte les règles que doivent respecter les entreprises et développeurs.

OWASP offre des ressources et des bonnes pratiques spécifiquement axées sur la sécurité des applications web, l'ANSSI établit des normes et des directives plus larges pour la sécurité des systèmes d'information dans leur ensemble

Nous utilisons **ZAP**, un logiciel open-sources de l'organisation OWASP. Permet la découverte des ouvertures liée tentative d'exploitation de vulnérabilités.

Les tests de sécurité entrepris avec ZAP :

- Évaluation de la vulnérabilité : Le système est analysé et analysé pour déceler les problèmes de sécurité.
- Tests de pénétration : Fait l'objet d'analyses et d'attaques de la part d'attaquants malveillants simulés.
- Tests d'exécution : L'application fait l'objet d'analyses et de tests de sécurité de la part d'un utilisateur final.
- Examen du code : Le code fait l'objet d'un examen et d'une analyse détaillés visant spécifiquement les vulnérabilités de sécurité.

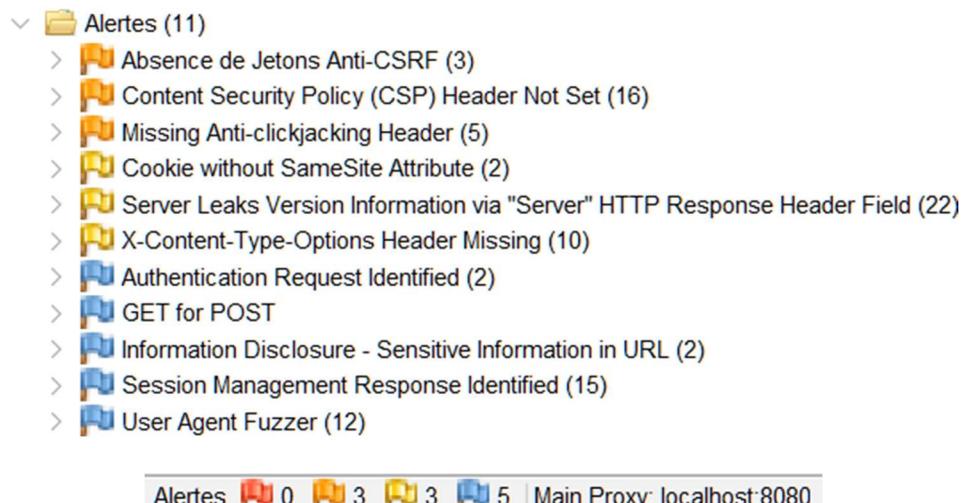


Figure 64 : Résultat test sécurité ZAP OWASP de l'applicatif web

On peut remarquer lors de l'analyse de sécurité l'absence d'alarmes critiques que représentent les *flags* rouges. Mais nous pouvons identifier trois *flags* oranges, ce qui indique qu'il y a quelques problèmes de sécurité que nous devons prendre en compte et corriger dès que possible pour améliorer la sécurité globale de notre application si cela nous semble nécessaire par rapport à notre utilisation finale. Les *flags* bleus sont des informations supplémentaires ou des conseils fournis par ZAP.

## 28 FINALITÉ DU PROJET WEB APP ARSn

En conclusion, le projet Web App ARSn représente une étape importante dans mon parcours de développement d'applications. Il a permis d'illustrer de manière concrète l'application des méthodologies de développement dans un contexte professionnel et informatique.

Chaque phase du développement des applicatifs ont été minutieusement explorés, de la conception à la réalisation en passant par les tests, offrant ainsi une vision globale du processus de création d'une application robuste et fonctionnelle pour le secteur industriel.

Les résultats obtenus sont représentés par une application lourde et une interface web, conçues sur mesure pour répondre aux besoins spécifiques de l'entreprise cible Stellantis (Fabriquant automobile). Leur évaluation a porté sur des critères de qualité, de sécurité et de performance.

Cependant, des axes d'amélioration ont été identifiés pour renforcer encore davantage la valeur ajoutée du projet. Il est notamment envisagé d'améliorer :

### Applicatif Lourd :

- L'esthétique de l'application de bureau (exécutable Applicatif Lourd)
- Gestion de session
- Archivage automatique de la base de données
- Procédure Readme de test de sécurité

### Applicatif Léger :

- L'esthétique de l'application Web (Applicatif Légère)
- Renforcement de sécurité XSS et mise en place de la méthode double authentification
- Permettre la portabilité mobile de l'application lourde
- Procédure Readme de tests de sécurité

En somme, le projet Web App ARSn constitue une étape cruciale dans l'évolution de mon expertise en développement d'applications.

## 29 COMPÉTENCES ACQUISES

La réalisation du projet Web App ARSN a été une opportunité enrichissante qui m'a permis d'acquérir un large éventail de compétences pertinentes dans le domaine du développement d'applications.

<u>Projet</u>	<u>Compétences</u>
Gestion de projet	-Mener à bien un projet de sa conception à son déploiement -Gestions des tâches
Analyse	-Analyse des besoins et des contraintes du projet
Développement d'une application web dite « Légère »	- Maîtrise des langages de programmation web (HTML, CSS, JavaScript) - Utilisation de Framework web (Flask, Peewee...) - Conception et développement d'interfaces utilisateurs responsives
Développement d'un exécutable dit « Lourd »	- Programmation en langages adaptés au développement d'applications lourdes (PyQt5) - Conception et développement d'applications robustes et performantes
Simulation Robotique couplé aux applicatifs externes	- Utilisation de logiciels de simulation robotique (RoboGuide) - Intégration et communication avec des applicatifs externes (APIs, protocoles de communication)
Sensibilisation sur la Sécurité et vulnérabilité	-Connaissance des principaux concepts de sécurité informatique - Identification des vulnérabilités courantes dans les applications web
Restitution d'un travail par un rapport technique rédigé	-Rédaction de rapports techniques clairs et concis -Présentation efficace des résultats et recommandations

## 30 CONCLUSION

Ce mémoire met en lumière l'importance cruciale de la cybersécurité dans le domaine de l'informatique et de la robotique, notamment dans le contexte de l'intégration de solutions robotisées dans l'industrie automobile. Alors que notre entreprise, ARSn, se positionne comme un acteur clé dans ce domaine en offrant des solutions innovantes et efficaces, il est essentiel de reconnaître les défis et les risques associés à la transformation numérique de nos systèmes de production.

En tant que futur diplômé du CDA et expert en numérisation industrielle, il est impératif de comprendre que la sécurisation des applications et des infrastructures informatiques est un élément fondamental de notre travail. Nous devons non seulement concevoir et développer des solutions technologiques avancées, mais également garantir leurs robustesses et leurs moyens de défenses contre les menaces cybernétiques.

La collaboration avec des références telles qu'OWASP et l'ANSSI peut être bénéfique pour notre entreprise, en nous fournissant des ressources et des directives pour renforcer notre posture de sécurité. En intégrant les meilleures pratiques de cybersécurité dès la phase de conception et en restant vigilants face aux menaces émergentes, nous pouvons assurer la fiabilité et la sécurité de nos solutions robotisées, tout en contribuant à la protection des données et à la préservation de la confiance de nos clients.

En définitive, ce mémoire souligne l'importance de la cybersécurité dans notre domaine d'expertise et appelle à une approche proactive et stratégique pour relever les défis de sécurité tout en continuant à innover et à transformer l'industrie automobile à travers des solutions numériques avancées.

## 31 LISTE DES FIGURES

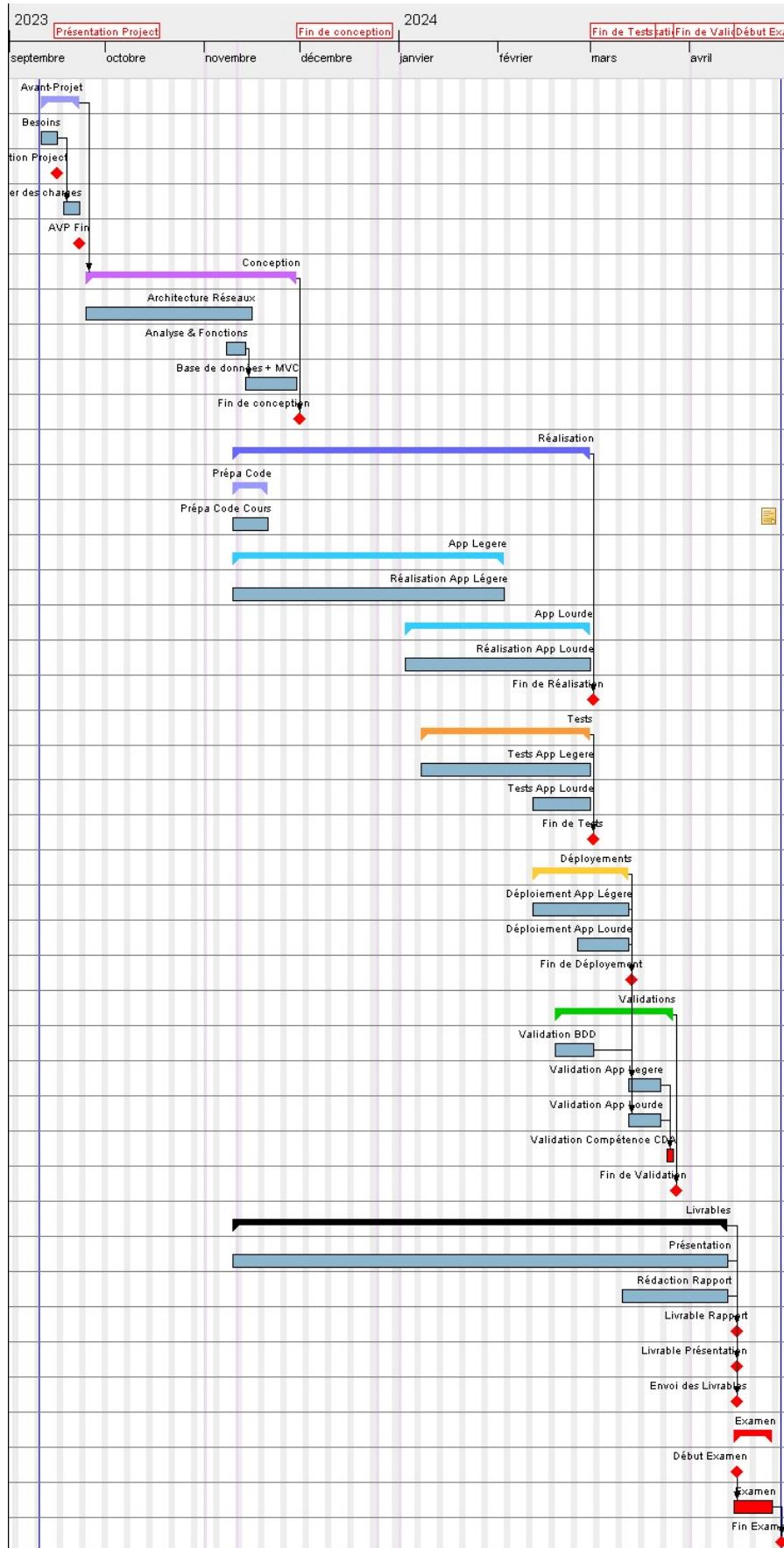
Figure 1 : La frise sales/collaborators depuis 10 ans .....	11
Figure 2 : Les secteurs & domaines d'activités .....	12
Figure 3 : Les Prestations et Activités de ARSn .....	13
Figure 4 : Clients satisfaits .....	13
Figure 5 : Organigramme simplifié de l'entreprise .....	14
Figure 6 : Logo du client .....	16
Figure 7 : Graphique historique de Stellantis .....	17
Figure 8 : Schéma implantation du système MEF-PAV (vue du dessus) .....	18
Figure 9 : Visualisation de la transition robotique vers un processus de persistance de données .....	19
Figure 10 : GANTTS & Planifications des tâches .....	21
Figure 11 : Illustration management Agile .....	21
Figure 11 : Structure du repository Github.....	22
Figure 12 : Branches Github.....	22
Figure 13 : Exemple d'un versionnage Github.....	22
Figure 14 : Vue AGV .....	23
Figure 15 : Vue robot .....	23
Figure 16 : Vue Tampon.....	23
Figure 17 : Vue Claye .....	23
Figure 18 : Schéma ligne montage de porte MEF .....	24
Figure 19 : Tableau des pièces primaires .....	25
Figure 20 : Tableau de la pièce finie .....	25
Figure 21 : Schéma fonctionnalités actuelle.....	26
Figure 22 : Schéma fonctionnalités proposées .....	26
Figure 23 : Schéma simplifié des échanges de données.....	27
Figure 24 : Diagramme de pieuvre .....	28
Figure 25 : Tableau des fonctions principales .....	29
Figure 26 : Tableau allocations des fonctions .....	30
Figure 27 : Figure cas d'utilisation des applicatifs.....	31
Figure 28 : Figure dictionnaire de classes & données .....	32
Figure 29 : Représentation des données du système du processus de fonctionnement .....	33
Figure 30 : Représentation des données du système affichage.....	33
Figure 31 : Représentation méthode MDL avec clés étrangères .....	34
Figure 32 : Prototypage & maquettage des pages web .....	35
Figure 33 : Illustration de l'architecture matériel de la solution proposé .....	36
Figure 34 : Schéma des 3 grandes couches réseau et interaction matériel .....	37
Figure 35 : Exemple de fournisseurs matériels.....	38
Figure 36 : Exemple de langage de programmation.....	38
Figure 37 : Page visualisation Alarme applicatif web .....	39
Figure 38 : Exemple du fichier de variables d'environnement «. env » .....	40
Figure 39 : Illustration capturé du site communautaire de l'OWASP : Recommandation TOP 10 .....	41
Figure 40 : Extrait du hachage de mot de passe utilisateur Application Légère .....	41
Figure 41 : Extrait fonction décoration « Brocken Access » des /route .....	41
Figure 42 : Extrait décorateur sur la route « /home ».....	41
Figure 43 : Page Requête applicatif lourd.....	42
Figure 47 : Extrait fonction « robot_move.py » .....	43
Figure 48 : Extrait programme « order_app.py ».....	44
Figure 44 : Interface de simulation RoboGuide .....	45
Figure 45 : Schématique de communication Robot serveur et AppLourde .....	46

Figure 46 : Interface Teach pendant du programme principal « A_ROBOT1 » .....	47
Figure 49 : Test unitaire connexion base de données SQL « test_if_bdd.py » .....	49
Figure 50 : Résultat du test unitaire « test_if_bdd.py » .....	49
Figure 51 : Test intégration authentification application « test_integration_auth.py » .....	50
Figure 52 : Résultat test intégration « test_integration_auth.py » .....	50
Figure 53 : Test fonctionnel connexion application web « test_fonctionnel_auth.py » .....	51
Figure 54 : Résultat test fonctionnel « test_fonctionnel_auth.py » .....	51
Figure 55 : Extrait Visual Code du fichier DockerFile .....	53
Figure 56 : Capture écran Portainer des stacks déployés .....	53
Figure 57 : Édition du stack « Flask_app_arsn » .....	54
Figure 58 : Capture écran Portainer de la liste des conteneurs déployés et en fonctionnement .....	54
Figure 59 : Journal (Logs) du conteneur « Flask_app_arsn » .....	55
Figure 60 : Test de charge application web « locustfile.py » .....	56
Figure 61 : Résultat interface Locust du test de charge « locustfile.py » .....	56
Figure 62 : Résultat graphique Locust du test de charge .....	57
Figure 63 : Résultat interface Chromium Lighthouse de l'applicatif web .....	57
Figure 64 : Résultat test sécurité ZAP OWASP de l'applicatif web .....	58

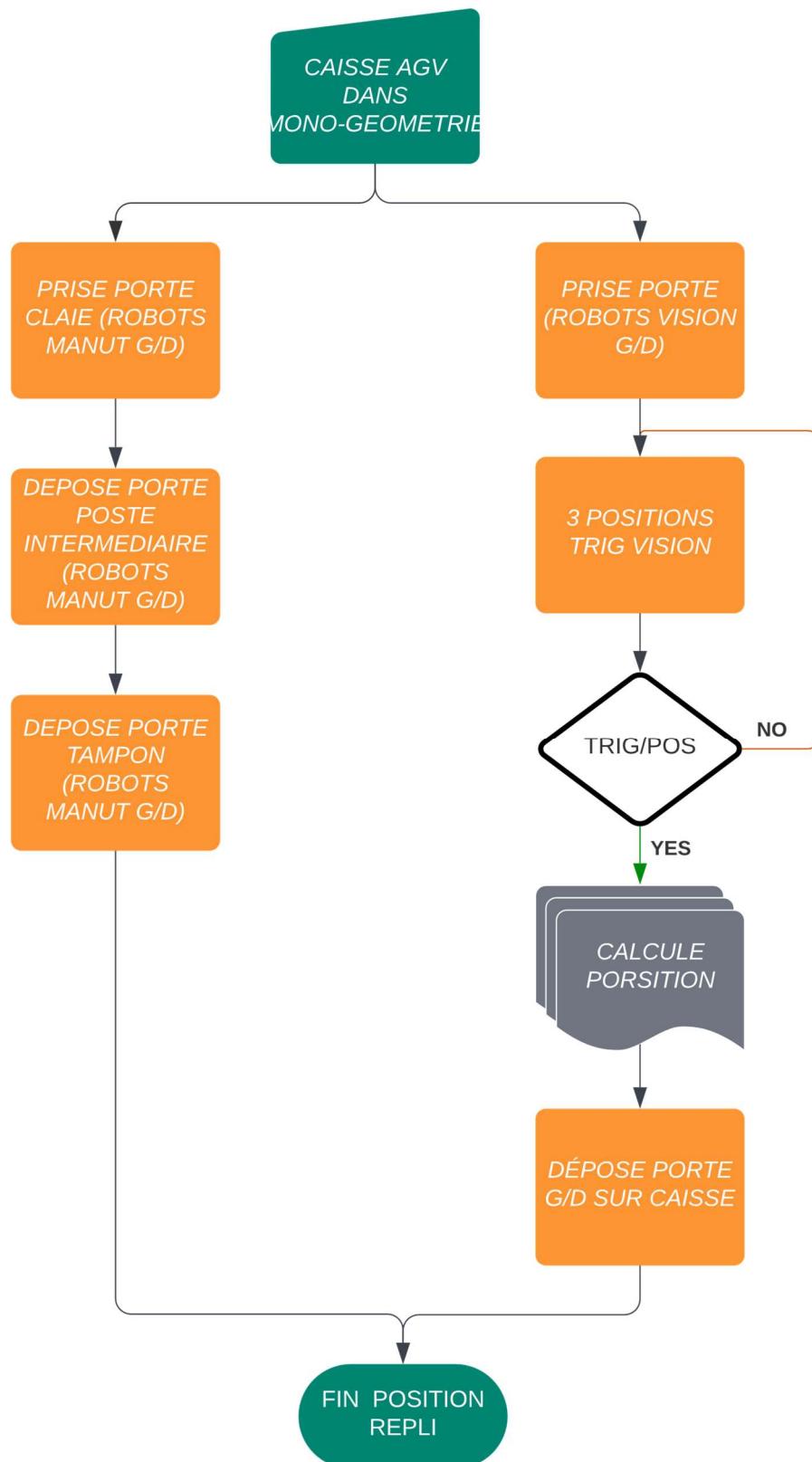
## 32 ANNEXES

GANTT développé .....	65
Cycle logigramme simplifié .....	66
Architecture de Développement .....	67
Architecture de Déploiement .....	68
Exemples de vues Applicatif Web .....	69
Exemples de vues Applicatif Lourd .....	70
Exemples de vues Teach Pendant .....	71
Extrait Phase1 Dictionnaire de données .....	72

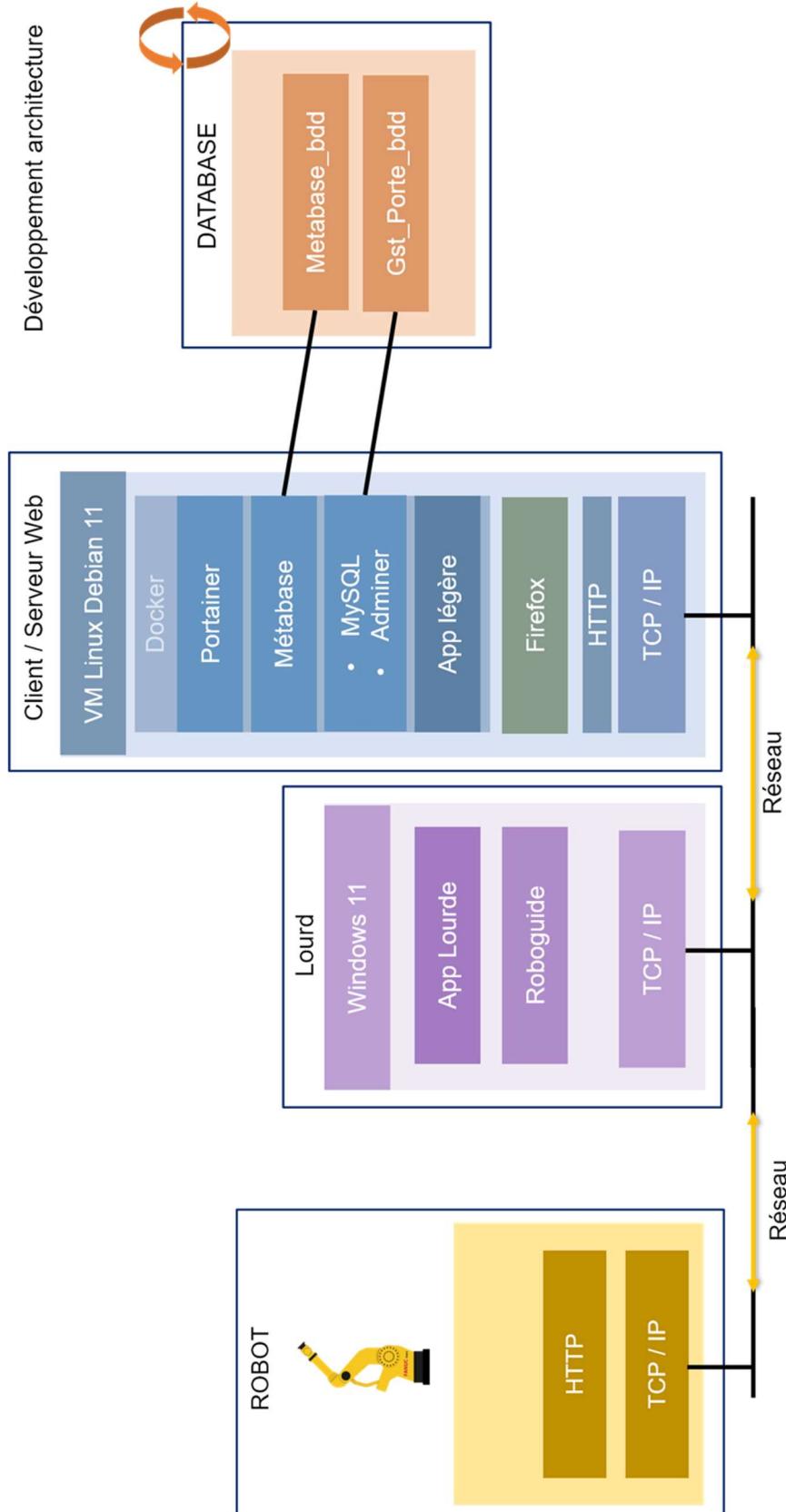
## GANTT développé



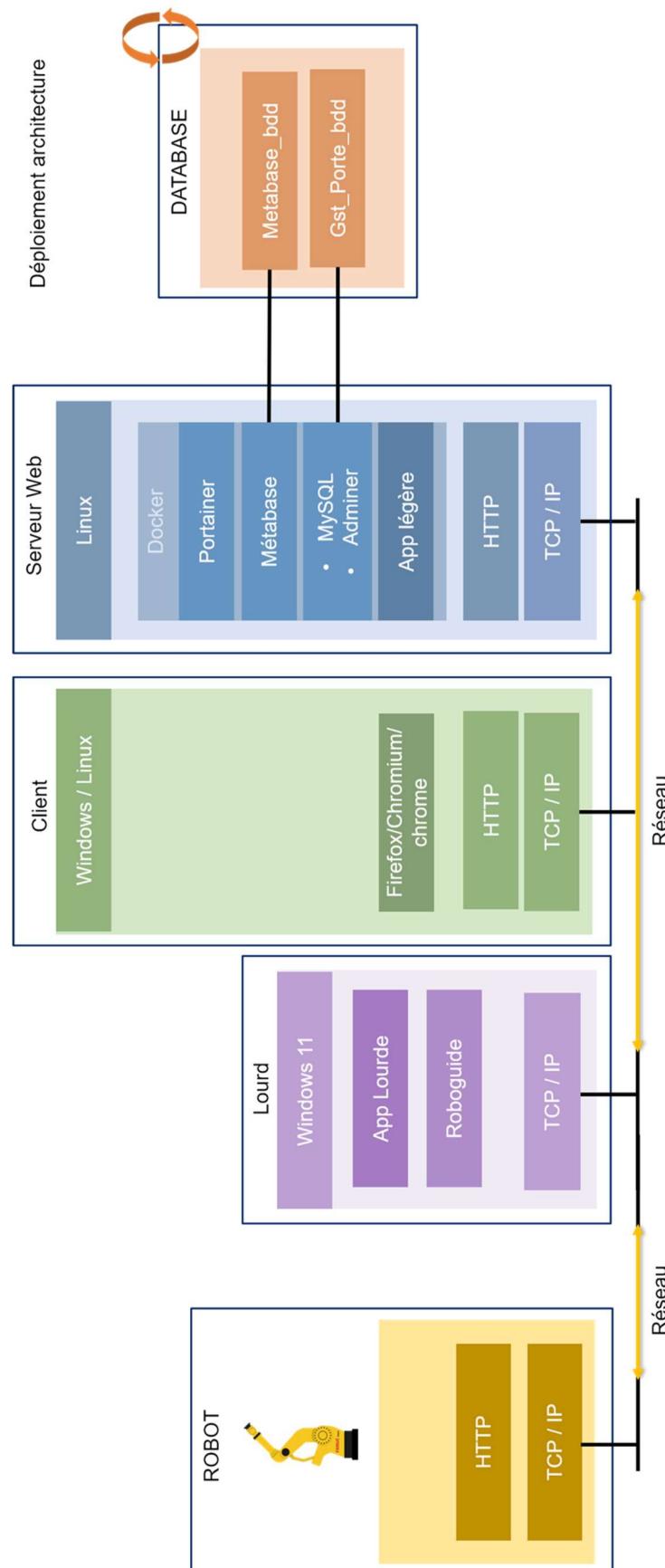
## Cycle logigramme simplifié



## Architecture de Développement



## Architecture de Déploiement



M4-CDA (Concepteur développeur d'applications)

## Exemples de vues Applicatif Web

Accueil Alarmes Statistiques Paramètres Appels CI

Statistiques & Courbes  
En développement.

Voiture

7 Total Voiture  
2 Distinct ID Option

How these Voiture are distributed

Marque	Nombre de lignes
Citroen	2
Ford	1
Peugeot	4

Modèle	Nombre de lignes
E-Méting	1
P87 500E	1
Smart	1
Zoe	1

ID Option - Couleur	Nombre de lignes
Vert	2
Jaune	3

ID Option - Tail-Overhead	Nombre de lignes
Vert	1

Propulsé par Metabase

2024 Stellantis-ARSn. Tous droits réservés.

Accueil Alarmes Statistiques Paramètres Appels CI

Cellule de Production  
En développement.

CONTROLE VISAGE LIENS AUTOMATES  
LIENS BASE DE DONNEE MOUVEMENT MANUEL DONNEES PORTES  
CHANGEMENT CAMPAGNE TESTS

© 2024 Stellantis-ARSn. Tous droits réservés.

Accueil Alarmes Statistiques Paramètres Appels CI

Contrôle PAV-G / ID  
En développement.

2023 Stellantis-ARSn. Tous droits réservés.

## Exemples de vues Applicatif Lourd

**App requettagé**

Order Cellule Utilisateur Web Archive

**Requettagé vers la table "utilisateur"**

Nom du l'utilisateur:

Mot de passe de l'utilisateur:

**Ajouter utilisateur**

Nom du l'utilisateur à modifier:

Nouveau nom du l'utilisateur:

Nouveau mot de passe de l'utilisateur:

**Modifier utilisateur**

**App requettagé**

Order Cellule Utilisateur Web Archive

**Visualisation Web**

**App requettagé**

Order Cellule Utilisateur Web Archive

**Requettagé vers la table "cellule"**

Nom Cellule:

Numero Ligne:

Batiment:

Secteur:

Ville:

Adresse

**Envoyer requête**

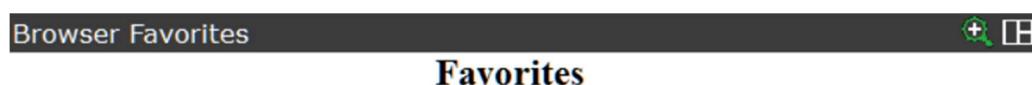
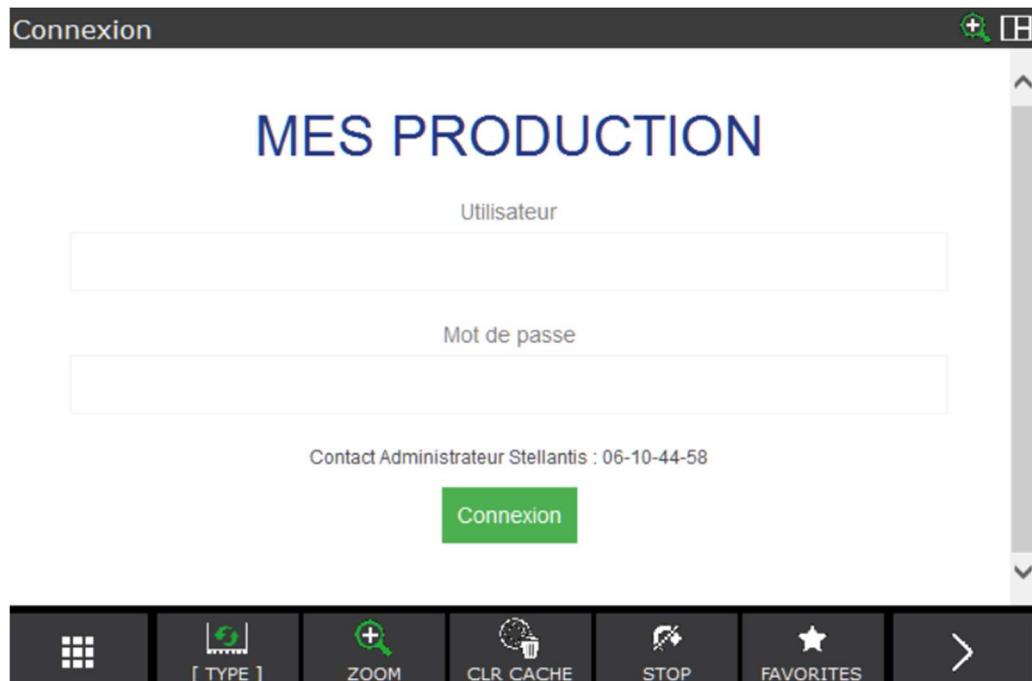
**App requettagé**

Order Cellule Utilisateur Web Archive

**Sauvegarde base de données "Gst\_Porte\_bdd"**

**Archiver la base de données**

## Exemples de vues Teach Pendant



[Browse ROBOT41 Home Page](#)

[Add a Link](#)

Browse Link	Modify Link Definition	Delete Link
<a href="http://172.23.32.117:5000/">ARSn APP (<a href="http://172.23.32.117:5000/">http://172.23.32.117:5000/</a>)</a>	<a href="#">Modify ARSn APP</a>	<a href="#">Delete ARSn APP</a>

## Extrait Phase1 Dictionnaire de données

Champ	Désignation	Exemple	Type	Taille	Remarque
nom_robot	nom du robot	R43	AN	5	
metier_robot	Manutention/Palettisation	MANUT	AN	255	
appli_robot	Vissage/Vision	Vissage	A	39	
v_appli	version application soft	V6.40			
modele_robot	modele fanuc	R2000ib-165F	A	39	
num_serie_robot	numero de serie unique	111111	A	39	
cmpt_heure_robot	compteur d'heure	279 000	N		
etat_robot	PROD/MANU	1/0	N		
date_achat	Mise en service	12/12/2012	N	10	
nom_visseuse	nom/id de la visseuse	V43	N	5	
modele_visseuse	modele visseuse	V9F	A	20	
num_serie_visseuse	numero de serie unique	11012	A	255	
couple_visseuse	couple visseuse Newton	32	A	255	
nb_vis	nombre de vissage	123	A	39	
nb_test_vis	nombre test vissage		AN	39	
test_couple	test couple Newton	22	AN	39	
cmpt_heure_visseuse	numero de heure unique	123	AN	39	
etat_visseuse	etat de la visseuse pendant le vissage	1 ou 2 ou 3	N	5	
date_visseuse	Mise en service	12/12/2012	N	10	
etat	etat visseuse PROD/MANU	0/1	AN	255	

AN = Alphanumérique

A = Alphabétique

N = Numérique